Inhaltsverzeichnis

1. Einleitung

1.1. Sinn und Zweck der Arbeit

Diese Diplomarbeit beschreibt die Planung, Entwicklung und Dokumentation einer Software, die in der Lage ist Seekarten zu lesen, diese Seekarten darstellen und mit GPS Daten zu kombinieren.

Diese Seekarten müssen nicht handelsübliche digitalisierte Seekarten sein, sondern können auch eingescannte Seekarten im JPEG Format sein.

Des weiteren ist die Software Plattformunabhängig und somit unter allen Betriebssystemen, die eine Unterstützung von Java ermöglichen, einsetzbar (Windows®, Linux, Macintosh®, usw.).

Aus Sicht der Informatik werden hier die Themenbereiche Softwareengineering, Mathematik und Implementierung berührt.

1.1.1. Softwareengineering

- Planung eines Softwareprojektes (Pflichtenhefterstellung)

- Durchführung eines Softwareprojektes (Implementation mit einer objektorientierten Programmiersprache, Testen der Software mit verschiedenen Methoden)

- Dokumentation eines Softwareprojektes

1.1.2. Mathematik

Zur Umrechnung von Pixelkoordinaten in geographische Koordinaten sowie für Kurs-und Distanzberechnungen werden hier folgende Elemente der Mathematik benötigt:

- Lineare Algebra (z.B. Gauß'sche Algorithmus)

- Trigonometrie (z.B. Winkelfunktionen)

1.1.3. Implementierung

Implementieren eines Softwareprojektes mit Hilfe der objektorientierten Sprachelemente von Java. Erstellen einer grafischen Benutzeroberfläche mit Java (Swing) anhand der Informationen und Anforderungen des Pflichtenheftes. Abspeichern von Datenelementen (Wegpunkte, Routen usw.) in XML-Dateien.

1.2. Wissenschaftlicher Anspruch

Der wissenschaftliche Anspruch dieser Diplomarbeit besteht in der Tatsache zu prüfen, ob es mit den Mitteln, die uns die objektorientierte Programmiersprache Java in der derzeitige verfügbaren Version zur Verfügung stellt, möglich ist, eine grafische Benutzeroberfläche für das Softwaretool GPSMan zu erstellen. Als besondere Anforderungen sind hier vor allem die Performance der Anwendung

im allgemeinen sowie die Darstellung u. U. sehr grosser Karten in einer akzeptablen Geschwindigkeit im Besonderen zu nennen.

Des weiteren ist zu prüfen, ob es mit den Mitteln von Java möglich ist, eingescannte JPEG Dateien als Seekarte zu kalibrieren, so dass ein realistischer Gebrauch auf einem Schiff möglich ist. Hiermit ist der Genauigkeitsfaktor hinsichtlich der Kalibrierung gemeint.

Zusätzlich ist die Vorgabe der Plattformunabhängigkeit zu beachten, um eine Software zu entwickeln, die nicht fest an ein bestimmtes Betriebssystem gebunden ist.

2. Aufgabenstellung

Die Aufgabenstellung der Diplomarbeit umfasst die Planung und Implementierung einer so genannten „Elektronischen Seekarte“. Mit diesem Softwaresystem sollen (Global Positioning System)-Daten verarbeitet und auf digitalisierten oder eingescannten Karten dargestellt werden. GPS Motion and Navigation (kurz **GPSMan**) soll eine Software sein, die unabhängig vom Betriebssystem und unabhängig von bestimmten Kartenformaten arbeiten kann.

Die Verwendung der Software soll den Schiffsführern von Segel- und Motorschiffen eine Hilfestellung bei der Navigation bieten. Einer der Kernpunkte ist die Verwendung von eingescannten Karten, womit der Benutzer der Software GPSMan nicht an Hersteller oder Distributoren digitalisierter Seekarten gebunden wird. Ein weiterer Kernpunkt ist die Implementierung in einer plattformunabhängigen Programmiersprache. Dadurch wird es dem Benutzer freigestellt seine verwendete Betriebssystemsplattform (Macintosh®, Linux oder Windows®) selbst zu wählen. Durch die Verwendung der Programmiersprache Java werden alle Plattformen unterstützt auf denen eine VM (Virtual Machine) von Java verfügbar ist.

Diese Diplomarbeit ist nicht die endgültige Form dieses Softwaresystems, sondern eher als Plattform oder Kernmodul für weitere an diesem Softwaresystem beteiligten Projekte zusehen. So ist z.B. die konkrete Implementierung von Schnittstellen für GPS Geräte oder das Logbuch nicht Bestandteil dieser Arbeit.

2.1. Bestandteile der Aufgabenstellung

- Implementierung einer GUI (Graphic User Interface, grafische Benutzeroberfläche) mit den entsprechenden Menüpunkten und einem Bereich um Images (Bilder, Seekarten) darstellen zu können.
- Einlesen und Darstellen von Karten im Format der Firma MAPTECH INC. (nachfolgend als KAP-Dateien bezeichnet)
- Einlesen und Darstellen von Bilder im JPEG Dateiformat
- Automatische Kalibrierung anhand der in der KAP-Datei vorhandenen Referenzpunkte (ein vereinfachtes Verfahren durch Auswahl der drei am weitesten voneinender entfernten Punkte, siehe Kapitel *Systementwurf / Berechnungsverfahren*)
- Manuelles Kalibrieren von Seekarten im JPEG Dateiformat.
- Das Erstellen, Darstellen und Verwalten von Wegpunkten und Routen mit der Maßgabe, dass Wegepunkte unabhängig von Routen und Routen unabhängig von Wegepunkten sind.
- Speichern und Laden von Wegepunktdateien im XML Dateiformat.
- Das Erstellen, Darstellen und Verwalten von Trackpunkten und Tracks.

- Speichern und Laden von Trackpunktdateien im XML Dateiformat.
- Die Simulation von GPS Daten (Schiffsbewegungen) nach vom Benutzer vorgegebenen Eckdaten.
- Das Messen von Distanzen mit Hilfe grafischer Werkzeuge in der Kartendarstellung.
- Das Berechnen von Kursen mit Hilfe grafischer Werkzeuge in der Kartendarstellung.

2.2. Spezielle Anforderungen aus Sicht der Informatik

Die Aufgabenstellung aus wissenschaftlicher Sicht eines Informatikstudenten betrachtet bietet folgende Anforderungen.

- Planung und Implementierung eines Softwareprojektes
    - Decodierung der KAP Dateien und Darstellung als Image in Java.
    - Implementierung einer Simulation von GPS Daten.
    - Verwaltung von Wegepunkte und Routen, in Anbetracht der gegenseitigen Unabhängigkeit.
    - Verwaltung von Trackpunkten und Tracks
- Testen eines Softwareprojektes
- Dokumentation eines Softwareprojektes

2.3. Anforderungen aus allgemeiner Sicht

Die Aufgabenstellung aus allgemeiner Sicht eines Studenten betrachtet bietet folgende Anforderungen.

- Einarbeitung in die Darstellung von nautischen Seekarten und deren Symbolen
- Einarbeitung in die Darstellungsformen von Karten (Projektionen und Ellipsoiden)
- Einarbeitung in die nautische Navigation von Schiffen zur Umsetzung von Messungen und Berechnungen im Softwaresystem GPSMan

3. Fachliches Umfeld

3.1. Grundbegriffe

Im diesem Kapitel werden Grundbegriffe derallgemeinen wissenschaftlichen Informatik sowie der nautischen Navigation erläutert, die im im Zusammenhang mit dieser Diplomarbeit stehen. Dies umfasst folgende Einzelthemen:

Grundbegriffe	***Erläuterung***
Kompressionsverfahren	Grundsätzliche Funktionsweise verschiedener Kompressionsverfahren mit Beispielen
Dateiformate	Grundsätzliche Informationen über Dateiformate
Bildformate	Grundsätzliche Informationen über Bildformate
Seekartendarstellungen	Erläuterungen der Begriffe Projektionen und Ellipsoiden
Nautische Navigation	Begriffserklärungen aus dem Umfeld der nautischen Navigation

3.1.1. Kompressionsverfahren

Die grundsätzliche Funktionsweise der Kompression (Komprimieren = verdichten, zusammendrücken) im Sinne der Informatik kann in wenigen Worten zusammengefasst werden. Hier die zwei fundamentale Methoden:

- Redundanzen (Wiederholungen) von Daten(-Strängen) werden erkannt und zusammengefasst (verkürzte Datenmodellierung)

- Unnötige und teilweise unwichtige Daten werden gelöscht. Die Löschung von Daten kann natürlich nur vorgenommen werden, wenn diese nicht mehr benötigt werden und auch später zur Verarbeitung unrelevant sind. Diese Art der Kompression wird z.B. für Video- oder Audiodaten verwendet.

Beide Prinzipien können selbstverständlich kombiniert werden.

Letztendlich ist Kompression auf verschiedene Bereiche des täglichen Lebens anwendbar. Speicherkapazität ist auch im Zeitalter von Gigabyte-Festplatte noch immer beschränkt und der Datentransfer im Internet fordert den Versand von Daten nicht in der Originalform, sondern im komprimierten Zustand (aus Zeit- und somit Kostengründen). Weiterhin ist Kompression für Backups sinnvoll, da ein Realtime-Zugriff nicht notwendig ist. Kompression sollte natürlich nur begrenzt in Bereichen eingesetzt werden, in denen ein besonders schneller Zugriff auf Daten notwendig ist, denn die Dekomprimierung fordert immer Rechenkapazität.

3.1.1.1. Übersicht der gängigen Verfahren

3.1.1.1.1. LZW-Codierung (Lempel, Ziv, Welch)

Es wird ein Wörterbuch aufgebaut (ein Wort wird durch ein Byte definiert) und

im Endeffekt wird für ein Folgewort nur noch ein Zeiger auf das erste Wort/das Wort im Wörterbuch gesetzt. Diese Kompression eignet sich natürlich sehr gut für Textdokumente, allerdings auch für Binärdaten, die viele gleiche Zeichenstränge beinhalten. LZW existiert mittlerweile in vielen Abwandlungen und in einigen sehr stark optimierten Formen. Die Kompression erfolgt verlustfrei, der Ursprungszustand des Dokumentes kann also wiederhergestellt werden.

3.1.1.1.2. RLE - Run Length Encoding

Einzelne gleiche Bytefolgen (z.B. 3, 3, 3, 3) werden zahlenmäßig zusammengefasst (z.B. 43, also 4 x 3) und nacheinander gespeichert. Diese Methode eignet sich hervorragend für Schwarz/Weiß Bilder oder für Bilder mit einer geringen Anzahl von Farben, sofern es sich nicht um ein gleichmäßiges Störbild handelt (also 1,0,1,0), da dann eine Vervielfältigung der Dateigröße vorgenommen wird. Auch diese Kompression ist verlustfrei.

3.1.1.1.3. Huffmann-Codierung

Alle Zeichen werden in einer Tabelle gespeichert. Die Tabelle wird so sortiert, dass häufig vorkommende Zeichen einen kürzeren Identifikationscode bekommen (z.B. ein oder zwei Bit) als ein nicht so häufig vorkommendes Zeichen (z.B. vier Bit). In Abwandelung kann auch eine Zeichenfolgentabelle gebildet werden. Die Datei wird so gespeichert, dass ein Abbild der Originaldatei gebildet wird, in der dann nur die über die Tabelle definierten Zeichen gespeichert werden (z.B. anstelle eines E dann ein Bit mit 0). Die Tabelle muss bei dieser Komprimierungsmethode mit in die Datei eingefügt werden, da sonst eine Dekomprimierung unmöglich ist. Dieses Vorgehen kann den Nachteil bergen, dass eine kurze Datei länger werden kann. Auch diese Komprimierungsform ist verlustfrei.

3.1.1.1.4. Wavelet-Kompression

Das Objekte/die Datei wird komplett betrachtet und mittels mathematischer Berechnung (Koeffizienten, Hoch- und Tiefpaßfilterung, etc.) gesplittet (non-blocking Strategie). Das genaue Verfahren ist allerdings für diese Arbeit zu komplex. Die Methode ist nicht verlustfrei (also nicht für Backups geeignet) und in der Folge der Berechnung wird meistens noch einmal per RLE oder Huffmann zusätzlich komprimiert. Diese Methode ist bestens für Bilder geeignet.

3.1.1.1.5. Fraktale-Kompression

Ein Bild wird in verschiedene Fragmente (geometrische Strukturen) geteilt und diese Fragmente werden dann durch pushing / turning / zooming / sizing verglichen. Es werden so genannte Domain- und Rangeblocks erzeugt (dynamisch). Zueinander ähnliche Blöcke werden gesucht; der Faktor der Ähnlichkeit kann eingestellt werden und hierüber bestimmt sich dann der Datenverlust. Jeder Domainblock wird mit mit jedem Rangeblock verglichen und außerdem wird die jeweilige Distanz zwischen den Domain- und dem transformierten Rangeblock ermittelt. Das Bild wird letztendlich durch eine Vielzahl mathematischer Gleichungen präsentiert. Eine Decodierung erfolgt durch Abarbeitung der Gleichungen und den Aufbau der Fragmente. Diese Komprimierung ist eigentlich nur für Bilder geeignet und auch nicht verlustfrei.

3.1.1.2. Übersicht der gängigen Datenformate

3.1.1.2.1. ZIP

Das ZIP-Format kombiniert diverse Komprimierungsverfahren (Huffmann, LZW, RLE und andere). Das ZIP-Format und der Einsatz der Kompressionsalgorithmen ist standardmäßig definiert.

3.1.1.2.2. KAP Komprimierung

Die in Dateien vom KAP-Format vorliegende Komprimierungsform ist von der MAPTECH INC. für die Komprimierung von eingescannten Seekarten entwickelt worden. Mithilfe der RLE Kodierung kann hier die Größe der Dateien entscheident verkleinert werden, da die Karten eine indizierte Farbcodierung mit maximal 128 Farben benutzen und die Karten große Flächen mit gleicher Farbe enthalten.

3.1.2. Dateiformate

(Quelle [IRL01])

Ein Dateiformat ist eine Richtlinie, die den Aufbau einer Sammlung von zusammengehörigen Daten, also einer Datei beschreibt. Sie legt fest, welche Bereiche der Datei für welchen Zweck dienen und in welcher Reihenfolge die Daten abgelegt werden. Einige Dateiformate haben sich im Laufe der Zeit als Standard durchgesetzt und werden weltweit eingesetzt.

3.1.2.1. XML Dateiformate

(Quelle [IRL01])

XML ist eine 1997 vom World Wide Web Konsortium (W3C) vorgestellte Beschreibungssprache, die eine reduzierte Variante von SGML darstellt. XML wurde als professionelle Alternative zu HTML konzipiert. In XML lassen sich eigene Befehle und Tags definieren. Sie wird häufig zur Darstellung von Datenbanken im Internet oder zur Abspeicherung von Dateninhalten genutzt.

3.1.3. Allgemeine Bildformate

3.1.3.1. Bitmap

Ein Bitmap ist ein Bild oder eine Grafik auf der Basis von Bits. Im allgemeinen sind mit Bitmaps Rastergrafiken gemeint, bei denen das Bild in unabhängig voneinander kontrollierbare Einzelpunkte aufgelöst wird. Die Höhe der Auflösung ergibt sich aus der Anzahl der Einzelpunkte innerhalb einer festgelegten Fläche. Die andere Variante der Bildspeicherung sind vektororientierte Grafiken oder Bilder, bei denen nicht einzelne Punkte, sondern mathematische Beschreibungen von allen im Bild vorkommenden geometrischen Figuren gespeichert werden.

3.1.3.2. JPG Bildformat

Abkürzung für "joint photografic expert group". Dieses Expertengremium erarbeitet seit 1988 international gültige Standards auf ISO und ITU-Basis für

JPEG und JBIG. Das Bildformat bietet ähnliche Merkmale wie GIF-Bilder, kann aber bis zu 16,7 Mio. Farben darstellen und unterliegt keinen Copyright-Merkmalen. Die JPEG-Kompression beschränkt sich nicht auf das Packen von Daten nach den üblichen Algorithmen, sondern beinhaltet raffinierte Verfahren, die selektiv einzelne Bildinformationen löschen, was bei sehr hohen Kompressionen zu störenden, rechteckig-verschachtelten Bildflecken - so genannten "Artefakten" - führen kann. Die Kompressionsrate ist in der Regel einstellbar.

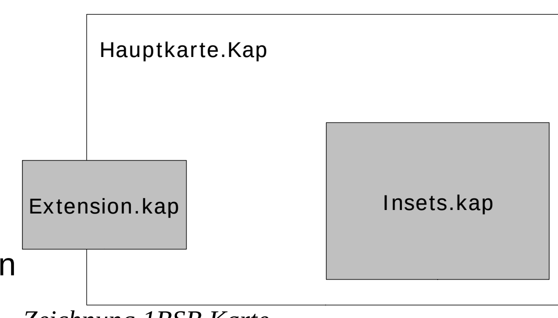
3.1.4. Seekartenbildformate

3.1.4.1. BSB Format

Das Seekartendatei im BSB Format ist eine „Navigationsdatei“. Sie regelt, welche KAP - Dateien benötigt werden und wie sie zusammen spielen. Außerdem beinhaltet sie Daten, die der Verwaltung dienen. Da viele Karten mehrere kleine Karten enthalten können, dient das BSB – File als eine Art „Hauptkarte“. In ihr werden alle enthaltenen Karten angegeben. Dabei unterscheidet man 3 Teile:

- Base (Hauptkarte)
- Inset (enthaltene Karte)
- Extension (enthaltene Karte, die über den Rand der Hauptkarte hinausreicht)

Jede dieser einzelnen Karten hat eine eindeutige Bezeichnung. Im folgenden Beispiel (Abb. 1) werden drei KAP – Dateien in der Struktur dargestellt. Die erste „Hauptkarte.kap“ stellt hierbei die Basis dar. Sie enthält die „Inset“ – Datei Insets.kap und die „Extension“- Datei Extension.kap.



Zeichnung 1BSB Karte

3.1.4.1.1. Aufbau

Der Aufbau einer BSB – Datei wird durch seine Signalwörter gekennzeichnet. Hierbei steht jedes Signalwort in der ersten Zeile und repräsentiert einen Datensatz. Zusätzlich ist es auch möglich, Kommentare anzugeben. Diese werden durch ein „!“ am Anfang gekennzeichnet. Jede Datei kann beliebig viele Kommentare enthalten, die bei der Bearbeitung nicht berücksichtigt werden. Hauptsächlich besteht aber eine BSB – Datei aus den bereits oben erwähnten Signalwörtern.
Diese Signalwörter können sein:

- **CHT**

Hier werden die generellen Kartenparameter beschrieben. Unterschieden wird dieses Feld nach

- NA: Name der Karte (max. 40 Charakter)
- NU: Nummerierung der Karte (max. 8 Charakter)

Diese beiden Felder werden auch später in der KAP – Datei aufgeführt.

- **CHF**

Dieses Signalwort steht für den Typ der Karte. In den meisten Fällen steht hier ein „HARBOR“ für eine Hafenkarte.

Mögliche Werte sind:

- Mineral, Training, Official, Plotting Sheet
- Harbor
- Coastal
- General
- Sailing, International
- Recreation
- Canoe
- Topographic
- Index
- Other

- **CED**

Das Wörtchen CED kennzeichnet die Version oder Ausgabe für die Kartenparameter. Unterschieden wird:

- *SE:* Ausgabe der NOAA, NIMA, CHS, USGS oder anderen Quellkarten.
- *RE:* Dieses Zeichnen kennzeichnet die Ausgabe der BSB–Dokumentation. Mit jeder neuen Ausgabe wird der Zähler um eins erhöht.
- *ED:* Erstellungsdatum

Alle diese Felder haben einen Größenbereich bis max. 8 Charakter.

- **VER**

Wie die Bezeichnung schon vermuten lässt, handelt es sich hier um die Version der Karte. Diese Versionsnummer spezifiziert das BSB–File–Format.

- **CHK**

An dieser Stelle wird die Anzahl, der für diese BSB – Karte verwendeten Karten, angegeben. Zudem kommt noch die Bezeichnung der einzelnen Karten. Der Aufbau gliedert sich folgendermaßen:

CHK/<Anzahl Karten>, <Karte 1>, ... , <Karte n>

- **CGD**

An dieser Stelle steht normalerweise der Küstendistrikt, in welchem Juristik – Bereich die Karte fällt.

- **ORG**

Das Kennzeichen definiert den Herausgeber der original Ausgabe. Mit „0“ wird angegeben, das es sich um die original Fassung handelt.

- **MFR**

Hier steht der Herausgeber dieser Karten. In fast allen Fällen wird es sich um „Maptech Inc.“ handeln.

- **K01 (bis Kn)**

Details der verwendeten Karten. Diese Details sind untergliedert in:

  - *NA:* Name der KAP - Datei.
  - *NU:* Nummerierung der KAP - Datei
  - *TY:* Typ der KAP – Datei. Wie bereits beim Aufbau beschrieben, können hier „BASE“,„INSET“, „EXTENSION“ stehen.
  - *FN:* Filename der KAP – Datei.

```
!Copyright 2001, Maptech Inc.  All Rights Reserved.
CHT/NA=LIPPE,NU=HLS01
CHF/HARBOR
CED/SE=2002,RE=1,ED=01/01/2002
VER/2.0
CHK/2,HLS01,HLS01_2
CGD/0
ORG/NAUTISCHE VEROFFENTLICHUNG
MFR/MAPTECH INC.
K01/NA=LIPPE,NU=HLS01,TY=BASE
    FN=HLS01_1.KAP
K02/NA=KRAMNITZE,NU=HLS01_2,TY=INSET
    FN=HLS01_2.KAP
```

Abbildung 1 BSB Datei

3.1.4.2. KAP Format

Hier folgt nun die ausführliche Beschreibung des KAP-Dateiformates.

3.1.4.2.1. Headerinformationen

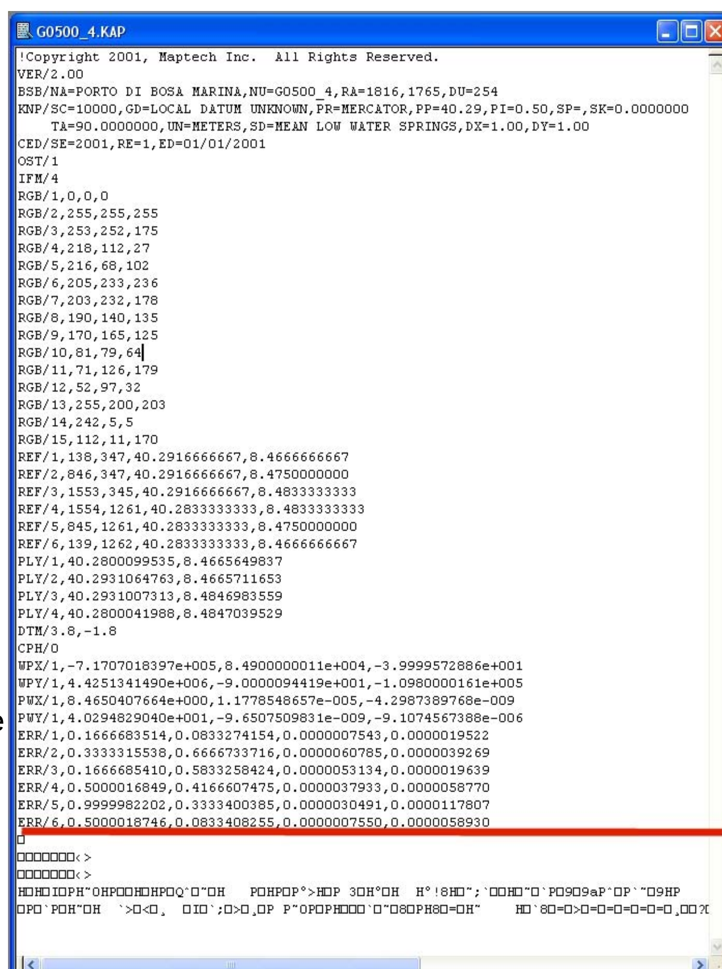
Die KAP-Datei beinhaltet alle Daten, die der Berechnung der Karte dienen. Anhand von diesen Daten, kann später im Programm die genaue Position bestimmt werden. Zusätzlich findet man hier auch Daten über die Farbgebung und verwendete Standards.

3.1.4.2.2. Der Aufbau

Der Aufbau der KAP-Dateien gliedert sich in zwei Teile

1. Die Daten, die für die Berechnung gebraucht werden und andere Parameterdaten bzgl. GPS

2. Die eigentlichen Bilddaten liegen in binärer Form vor



```
G0500_4.KAP
!Copyright 2001, Maptech Inc.  All Rights Reserved.
VER/2.00
BSB/NA=PORTO DI BOSA MARINA,NU=G0500_4,RA=1816,1765,DU=254
KNP/SC=10000,GD=LOCAL DATUM UNKNOWN,PR=MERCATOR,PP=40.29,PI=0.50,SP=,SK=0.0000000
    TA=90.0000000,UN=METERS,SD=MEAN LOW WATER SPRINGS,DX=1.00,DY=1.00
CED/SE=2001,RE=1,ED=01/01/2001
OST/1
IFM/4
RGB/1,0,0,0
RGB/2,255,255,255
RGB/3,253,252,175
RGB/4,218,112,27
RGB/5,216,68,102
RGB/6,205,233,236
RGB/7,203,232,178
RGB/8,190,140,135
RGB/9,170,165,125
RGB/10,81,79,64
RGB/11,71,126,179
RGB/12,52,97,32
RGB/13,255,200,203
RGB/14,242,5,5
RGB/15,112,11,170
REF/1,138,347,40.2916666667,8.4666666667
REF/2,846,347,40.2916666667,8.4750000000
REF/3,1553,345,40.2916666667,8.4833333333
REF/4,1554,1261,40.2833333333,8.4833333333
REF/5,845,1261,40.2833333333,8.4750000000
REF/6,139,1262,40.2833333333,8.4666666667
PLY/1,40.2800099535,8.4665649837
PLY/2,40.2931064763,8.4665711653
PLY/3,40.2931007313,8.4846983559
PLY/4,40.2800041988,8.4847039529
DTM/3.8,-1.8
CPH/0
WPX/1,-7.1707018397e+005,8.4900000011e+004,-3.9999572886e+001
WPY/1,4.4251341490e+006,-9.0000094419e+001,-1.0980000161e+005
PWX/1,8.4650407664e+000,1.1778548657e-005,-4.2987389768e-009
PWY/1,4.0294829040e+001,-9.6507509831e-009,-9.1074567388e-006
ERR/1,0.1666683514,0.0833274154,0.0000007543,0.0000019522
ERR/2,0.3333315538,0.6666733716,0.0000060785,0.0000039269
ERR/3,0.1666685410,0.5833258424,0.0000053134,0.0000019639
ERR/4,0.5000016849,0.4166607475,0.0000037933,0.0000058770
ERR/5,0.9999982202,0.3333400385,0.0000030491,0.0000117807
ERR/6,0.5000018746,0.0833408255,0.0000007550,0.0000058930
```

Abbildung 2 KAP Datei

Die Bilddaten werden hier in der „National Ocean Service“ NOS Kompression angegeben. Diese Kompression eignet sich besonderes für die Kartenabbildungen. Der Aufbau der KAP-Dateien richtet sich nach Signalwörtern. Die Gliederung beginnt auch hier mit dem Copyright der Fa. Maptech (das als Kommentar angegeben ist) die die Rechte an den digitalisierten Karten besitzt.

Hier die anderen Signalwörter:

- **VER**

  Es handelt sich hier um die Version der Karte.

- **BSB**

  Das Signalwort BSB kennzeichnet die Verwaltungsdaten der Karte. Es untergliedert sich in:

  - **NA**

    Name der Karte (z.B. Elba). Der Wert darf max 40 Zeichen enthalten.

  - **NU**

    Nummerierung der Karte (z.B. G0434). In der Regel ist das auch der Dateiname der Karte. Hier sind max. 8 Zeichen erlaubt.

  - **RA**

    Breite und Höhe der Karte in Pixel (z.B. 11693,8395). Hier sind max. 11 Zeichen erlaubt.

- **DU**

  Die Maßeinheiten der Zeichnung in pixel / inch.

- **KNP**

An dieser Stelle stehen die Daten, die für die Projektion der Karte gebraucht werden. (Näheres im Unterkapitel 3.3)

- **CED**

  Dieser Block enthält zusätzliche Verwaltungsinformationen für die Karte. Sie sind untergliedert in:

  - **SE:** Ausgabe der NOOA, NIMA, CHS, USGS oder anderen Quellkarten

  - **RE:** Kennzeichnet die Ausgabe der BSB Dokumentation. Mit jeder neuen Ausgabe wird diese Zahl um eins erhöht.

  - **ED:** Erstellungsdatum

  Alle dieser Felder haben einen Größenbereich von max. 8 Zeichen.

- **OST**

  OST definiert eine Anzahl von Bildzeilen zwischen den Offset-Werten des Bildes.

- **IFM**

  Dieses Kennzeichen definiert die NOS-Kompression für die Bilddatei. Folgende Typen werden unterstützt:

  - 1 NOS Kompression Type 1, 1 bit color (Bilevel, 2 colors)

  - 2 NOS Kompression Type 2, 2 bit color (2 to 3 colors)

  - 3 NOS Kompression Type 3, 3 bit color (4 to 7 colors)

  - 4 NOS Kompression Type 4, 4 bit color (8 to 15 colors)

  - 5 NOS Kompression Type 5, 5 bit color (16 to 31 colors)

  - 6 NOS Kompression Type 6, 6 bit color (32 to 63 colors)

  - 7 NOS Kompression Type 7, 7 bit color (64 to 127 colors)

    (Mehr zur Bildkompression erfahren Sie im Kapitel 3.8)

- **RGB**

  Dieses Signalwort kennzeichnet die Farben für die Karte. Weitere Informationen werden im dem Unterkapitel 3.4 behandelt.

- **REF**

  Unter REF werden die Referenzpunkte für die Karte angegeben. Informationen zur Berechnung und Funktionalität dieser Punkte finden Sie unter Unterkapitel 3.5.

- **PLY**

  Mit PLY werden die Eck- und Randpunkte der Karte angegeben. Im Unterkapitel 3.6 finden Sie weitere Informationen hierzu.

- **CPH**

  CPH beschreibt einen Wert für eine Phasenverschiebung. Diese wird in Karten benötigt, die sich über den Null-Meridian erstrecken. Ein positiver Wert für CPH hat den Effekt, dass die Koordinaten mathematisch ununterbrochen weitergehen. CPH wird dabei für Längengrade gebraucht, die ein ununterbrochenes mathematisches Fortschreiten voraussetzen. Der Aufbau von CPH gliedert sich wie folgt:

  CPH/ [Wert]

  Hierbei ist [Wert] eine positive reale Zahl von 0 bis 180°. Dabei steht die 0 für Karten, die den Null Meridian nicht überspannen. Für die anderen Karten ist CPH gleich der absolute Wert von dem am östlich gelegenem PLY Punkt in der westlichen Hemisphäre der Karte. Fügt man CPH einer Karte zu, so hat das den Effekt, das ihre Koordinaten als mathematisch ununterbrochen angesehen werden. Im Gegensatz dazu, führt CPH bei Karten aus der westlichen Hemisphäre mit negativen Koordinaten dazu, das die richtigen verschobenen Koordinaten gezeigt werden.

- **WPX**

  WPX definiert ein, auf den oben beschriebenen REF Zeilen, basierendes Polynom. Das Polynom benutzt Welt Breiten- und Längengrade (X_m, Y_m) um ein übereinstimmendes X Bild (X_i) Zeilenwert in der Bilddatei zu berechnen. Die polynomische Funktion ist wie folgt definiert:

  $$X_i = A+Bx_m+Cy_m+Dx_m^2+Ex_my_m+Fy_m^2+Gx_m^3+Hx_m^2y_m+Ix_my_m^2+Jy_m^3+\ldots$$

  Hierbei ist Xi die Position des Bildpixels x auf dem Breiten- und Längengrad, welcher als dezimaler Grad eingeben wurde.

  Der Aufbau der WPX Zeile gliedert sich wie folgt:

WPX/n, A, B, C, D, E, F, ... , J

A bis J sind dabei die polinomischen Koeffizienten. „n“ ist die Anzahl der polinomischen Koeffizienten dar und beträgt maximal 3, je nachdem, wie viele gebraucht werden um das genaueste Ergebnis zu erzielen.

Wenn n=1, dann werden die Terme A bis C benutzt

Wenn n=2, dann werden die Terme A bis F benutzt

Wenn n=3, dann werden die Terme A bis J benutzt.

- **WPY**

Das Polynom benutzt Welt Breiten- und Längengrade (X_m, Y_m) um ein übereinstimmendes y Bild (Y_i) Spaltenwert in der Bilddatei zu berechnen. Die polynomische Funktion ist wie folgt definiert:

$$Y_i = A+Bx_m+Cy_m+Dx_m^2+Ex_my_m+Fy_m^2+Gx_m^3+Hx_m^2y_m+Ix_my_m^2+Jy_m^3+...$$

Hierbei ist X_i die Position des Bildpixels x auf dem Breiten- und Längengrad, welcher als dezimaler Grad eingeben wurde.

Der Aufbau der WPY Zeile gliedert sich wie folgt:

WPY/n, A, B, C, D, E, F, ... , J

A bis J sind dabei die polinomischen Koeffizienten. „n“ ist die Anzahl der polinomischen Koeffizienten dar und beträgt maximal 3, je nachdem, wie viele gebraucht werden um das genaueste Ergebnis zu erzielen.

Wenn n=1, dann werden die Terme A bis C benutzt

Wenn n=2, dann werden die Terme A bis F benutzt

Wenn n=3, dann werden die Terme A bis J benutzt.

- **PWX**

PWX definiert ein, auf den oben beschriebenen REF Zeilen, basierendes Polynom. Das Polynom benutzt die Bildpixel (xi,yi) um einen übereinstimmenden Längengrad zu berechnen. Die polynomische Funktion ist wie folgt definiert:

$$X_m = A+Bx_i+Cy_i+Dx_i^2+Ex_iy_i+Fy_i^2+Gx_i^3+Hx_i^2y_i+Ix_iy_i^2+Jy_i^3+ ...$$

X_m ist hier der berechnete Breitengrad bei den Bildpixeln (xi,yi), welcher als dezimaler Grad angegeben wird. Der Aufbau der PWX Zeile gliedert sich wie folgt:

PWX/n, A, B, C, D, E, F, ... , J

A bis J sind dabei die polinomischen Koeffizienten. „n“ ist die Anzahl der polinomischen Koeffizienten dar und beträgt maximal 3, je nachdem, wie viele gebraucht werden um das genaueste Ergebnis zu erzielen.

Wenn n=1, dann werden die Terme A bis C benutzt

Wenn n=2, dann werden die Terme A bis F benutzt

Wenn n=3, dann werden die Terme A bis J benutzt.

- **PWY**

PWY definiert ein, auf den oben beschriebenen REF Zeilen, basierendes Polynom. Das Polynom benutzt die Bildpixel (xi,yi) um einen übereinstimmenden Breitengrad zu berechnen. Die polynomische Funktion ist wie folgt definiert:

$$X_m = A+Bx_i+Cy_i+Dx_i^2+Ex_iy_i+Fy_i^2+Gx_i^3+Hx_i^2y_i+Ix_iy_i^2+Jy_i^3+\ldots$$

X_m ist hier der berechnete Breitengrad bei den Bildpixeln (xi,yi), welcher als dezimaler Grad angegeben wird.

Der Aufbau der PWY Zeile gliedert sich wie folgt:

PWY/n, A, B, C, D, E, F, ... , J

A bis J sind dabei die polinomischen Koeffizienten. „n“ ist die Anzahl der polinomischen Koeffizienten dar und beträgt maximal 3, je nachdem, wie viele gebraucht werden um das genaueste Ergebnis zu erzielen.

Wenn n=1, dann werden die Terme A bis C benutzt

Wenn n=2, dann werden die Terme A bis F benutzt

Wenn n=3, dann werden die Terme A bis J benutzt.

- **ERR**

In der Zeile ERR stehen die berechneten Fehler im Bezug auf die REF Punkte, die entstehen wenn X_i, Y_i or X_m, Y_m anhand der Polynome aus PWX, PWY oder WPX, WPY kalkuliert werden. Die Fehlerdaten sollen dem Kartenbenutzer behilflich sein um die Abweichungen zwischen der aktuellen geographischen Position und der angezeigten Grafischen Position abschätzen zu können. Diese Fehler können dann z.B. in die Berechnung für das Intervall zum auslösen des Alarms einfließen (z.B. größeres Intervall für den Alarm beim geringen Abstand zum Hindernis).

Die ERR Zeile wird wie folgt aufgebaut:

ERR/[Point Number], X_iError, Y_iError, Y_mError, X_mError

Unter [Point Number] muss die Nummer der zugehörigen REF Zeile stehen.

X_iError ist der X Pixelfehler, der aus der Berechnung mit WPX, in Pixeln entsteht.

Y_iError ist der Y Pixelfehler, der aus der Berechnung mit WPY, in Pixeln entsteht.

Y_mError ist der y Breitengradfehler, der aus der Berechnung mit PWY, in der Einheit Dezimalgrad entsteht.

X_mError ist der x Längengradfehler, der aus der Berechnung mit PWX, in der Einheit Dezimalgrad entsteht.

3.1.4.2.3. Signalwort KNP

Wie bereits oben kurz beschrieben, steht das Signalwort KNP für die Daten der Projektion. Hier wird bestimmt, welcher Ellipsoid und welche Projektionsart für die Karte verwendet werden. Hierbei wird der Ellipsoid (z.B. WGS84) benötigt um die Position richtig zu berechnen. Die Projektionsart (z.B. Mercator) wird benötigt um die Position in der Karte richtig ein zu zeichnen. Diese beiden Begriffe sind auch im Kapitel Begriffserklärung näher erläutert.

Das Signalwort gliedert sich in folgende Teilbereiche:

- **SC**

  SC gibt den Maßstab der Karte wieder. Nehmen wir z.B. einen Maßstab von 3000. Diese 3000 bedeutet einen Maßstab von 1:3000 (Angabe in cm)

- **GD**

  Hier steht nun der schon oben angesprochene Ellipsoid. Beispiele sind der Ellipsoid „WGS84“ oder der von Bessel (siehe Begriffserklärung).

- **PR**

  Unter diesem Punkt steht die Art der Projektion. Als Beispiele kann man hier die Mercator oder die Gauß-Krüger Projektion nennen (siehe Begriffserklärung).

- **PP**

  Dieser Punkt beschreibt den Mittelpunkt des Mercator Zylinders. Dieser Wert wird in der Form Grad.Grad angegeben. Z.B kann der Wert 54.71 verwendet werden. Das bedeute 54° Nord und 42,6 Ost (0,71 * 60). Dieser Punkt liegt in den meisten Fällen in der Mitte der Karte. Es ist aber auch Möglich, das dieser Punkt außerhalb der Karte liegt bzw. nach Norden oder Süden

verschoben ist.

- **PI**

  PI beschreibt den Abstand der einzelnen parallelen Längen- und Breitengrade. Die Angabe erfolgt in Minuten.

- **SP**

  Flaches Koordinatensystem und Intervall in US feet. Wenn es mehr als ein Koordinatensystem auf der Karte gibt, wird jedes System in der Form SP1, Intervall, SP2, Intervall, ... angegeben.

- **SK**

  Mit diesem Wert wird der Grad der Kartendrehung beschrieben. Das heißt, das dieser Wert angibt wie weit die Karte gedreht werden muss, damit sie Richtung Norden zeigt. Zur Hilfe beinhalten manche Karten eine Kompassabbildung. Die Angabe ist wie im Punkt PI in dezimalen Gradeinheiten angegeben. Die Drehung der Karte erfolgt im Uhrzeigersinn. Der Wert kann bis zu max. 6 Zeichen groß sein. Max. Wert ist dabei 359,99°.

- **TA**

  Der von der oberen linken Ecke des Bildes im Uhrzeigersinn zu einem Vektor, der annähernd parallel zur überwiegenden Ausrichtung des Tests steht, gedrehte Winkel. Normalerweise ist der Wert für eine Karte, mit einem ursprünglichem Entwurf für north-up Ausrichtung, 90° (SK=0) .

- **UN**

  Hier wird die Einheit der Tiefenangaben beschrieben.

- **SD**

  Beschreibt das Format für die Tiefenlotung.

- **DX , DY**

  Diese Werte beschreiben die Anzahl von Metern für einen Pixel. Anhand dieser Angaben kann die Breite, Länge der Karte und die Entfernungen auf der Karte bestimmt werden. Hierzu werden allerdings zusätzlich die Daten von RA aus dem Signalwort BSB benötigt.

  Berechnung anhand von Beispieldaten

  - **RA**: 1100, 1550 (Länge (py), Breite (px))
  - **DX**: 0.30 (Meter / Pixel)

- **DY**: 0.30 (Meter / Pixel)

  Länge = py * DY = 1100 * 0,30 = 330 Meter

  Breite = px * DX = 1550 * 0,30 = 465 Meter

3.1.4.2.4. Signalwort RGB

Das Signalwort RGB kennzeichnet eine Standard Farbpalette, welche benutzt wird um die Bilddatei zu präsentieren. Hierbei muss jede KAP-Datei mindestens zwei RGB Zeilen enthalten.

Der Aufbau für RGB gliedert sich immer wie folgt:

RGB [Index], [Rot-Anteil], [Grün-Anteil] [Blau-Anteil]

Der Wertebereich für die Farbanteile liegt zwischen 0 und 255. Diese Werte gelten für die Indoor Variante, welche auch die Standardeinstellung ist. Daneben gibt es Farbpaletten für die Tagansicht, die Nachtansicht und die Ansicht bei Nebel. Der Unterschied bei den Farbpaletten liegt lediglich in der Umgebung, in welcher die Karte betrachtet wird.

- **DAY**

  Die Farbpalette für die Tagansicht wird unter einem optionalem Signalwort DAY angegeben.

- **DSK**

  Genauso wie bei DAY gibt es auch hier ein Signalwort für die Ansicht bei Nebel. Dieses Signalwort wird dann als DUSK bezeichnet.

- **NGT**

  Die Nachtansicht ist wie bereits die anderen Ansichten optional und definiert einen Farbbereich für eine Nachtumgebung.

3.1.4.2.5. Signalwort REF

Das Signalwort REF definiert einen Bezugspunktparameter, dieser kann benutzt werden um die Koeffizienten für eine Vielzahl von polynomisch mathematischen Modellen. Diese Koeffizienten können benutzt werden um geographische Daten (Längengrade, Breitengrade) in Bildpixel Koordinaten umzuwandeln. Entwickler können diese Punkte benutzen um ihre eigenen Umwandlungskoeffizienten zu berechnen. Andere andere Möglichkeit hierzu stellen die Signalwörter WPX, WPY, PWX und PWY dar, die bereits vorher behandelt wurden.

Die Anzahl der REF Umwandlungspunkte ist in ihrem Maximum unbeschränkt. Mindestanzahl sind 2 REF Punkte.

Der Aufbau dieser Punkte richtet sich immer nach folgendem Schema:

REF [Index], [X-Bild], [Y-Bild], [Breitengrad], [Längengrad]

Die [X-Bild] und [Y-Bild] Koordinate beschreibt einen Pixelpunkt innerhalb des Bildes. Die Koordinaten für den Breitengrad bzw. Längengrad entsprechen den tatsächlichen geographischen Daten.

3.1.4.2.6. Signalwort PLY

PLY definiert einen polinomischen Rahmen um den Navigationsbereich der Karte. Jede dieser PLY Zeilen entspricht einem geographischem Punkt auf der Karte und wird folgendermaßen aufgebaut:

PLY/[Point Number], [Breitengrad], [Längengrad]

Verbindet man die Punkte mit Linien, so stellen sie die Start- und Endpunkte der Karte dar. Die Anzahl dieser Rahmenpunkte ist nicht eingeschränkt und erlaubt so eine ziemlich genaue Darstellung des Rahmens.

3.1.4.2.7. Die Bilddaten

Die Bilddatensektion beginnt mit einem einzigen binärem Byte, welches die Kompressionstyp darstellt. Dieser Wert sollte gleich dem unter IFM definiertem Wert sein. Die binären Bilddaten sind komprimiert und stellen eine line-by-line Typ dar, wenn OST = 1 ist.

Die binären Bilddaten sind innerhalb der KAP-Datei in der NOS Kompression gespeichert. Diese Kompression ist als ein Bit Stream mit Byte-Grenzen codiert. Eine Einheit von Informationen benutzt soviel Bytes, wie es für seine Größe benötigt. Die 7 least significant bits (LBS) Bits sind dabei der Wert. Die most significant bits (MSB) stellen ein Konkatenationszeichen dar. Wenn das high order bit eins ist, dann benötigt man für den Wert eine Konkatenation mit dem nächstem Byte. Es ist analog zu der ASCII Repräsentation von numerischen Daten. Jedes Byte des Dezimalstrings erhöht die Größe um den Faktor 10. Jedes Byte der NOS Kompression erhöht die Größe um den Faktor 128.

Jede Zeile der gerasterten Bilddaten ist wie folgt codiert:

- LineNumber Color 1, Run 1 ... Color n, Run n NULL

  LineNumber = Sequential line count encoded in the MSB concatenation scheme. Top line = 1.

  Color, Run: Coded in accordance with the format selection, in the NOS compression scheme. The sum of the runs on a line = the line width.

  NULL: Binary 0 value used to signify line termination.

Die folgenden Color/Run bit stream kombinationen sind möglich:

- 1 NOS Kompression Type 1, 1 bit color (Bilevel, 2 colors)
- 2 NOS Kompression Type 2, 2 bit color (2 to 3 colors)

- 3 NOS Kompression Type 3, 3 bit color (4 to 7 colors)
- 4 NOS Kompression Type 4, 4 bit color (8 to 15 colors)
- 5 NOS Kompression Type 5, 5 bit color (16 to 31 colors)
- 6 NOS Kompression Type 6, 6 bit color (32 to 63 colors)
- 7 NOS Kompression Type 7, 7 bit color (64 to 127 colors)

Beispiel 1: 2 Byte im Format 3; F6 70 (H) = 11110110 01110000 (B)

Byte 1 Flag bit 1 Concatenate the next byte

Format 3: 3 Color bits 1 1 |

2 1 |

3 1 |Color is palette number 7

Run Length bits 1 0 |

2 1 |

3 1 |

4 0 |Byte 1 portion of run is 6

Byte 2 Flag bit 0 Final byte of the stream

Run Length continues 5 1 |

6 1 |

7 1 |

8 0 |

9 0 |

10 0 |

11 0 |Byte 2 part of run is 112

Die run length ist (Byte 1 Run) * 128 + (Byte 2 Run) + 1 = 881. Wenn das eine 3 Byte Konkatenation ist, würde die run length : (Byte 1 Run)*128*128+(Byte 2 Run)*128+(Byte 3 Run)+1 betragen.

Beispiel 2: Voraussetzung ist eine Zeile als Byte-String: 01 F6 70 76 00

01: Y = 1 Color/Runs interpreted at the different formats:

Type	Color 1	Run 1	Color 2	Run 2
2	3	2929	3	23
3	7	881	7	7
4	14	881	14	7
5	29	369	29	3
6	59	113	59	1

00: End of Line

3.1.5. Seekartendarstellungen

In der Darstellung von Seekarten gibt es verschiedene Formen der Projektionen und der dargestellten Ellipsoiden. Mit den folgenden Informationen soll ein Grundverständnis der herrschenden Problematiken zur Darstellung unserer Erdoberfläche vermittelt werden.

3.1.5.1. Ellipsoide

(Quelle [LAB01])

Eine Position auf der unregelmäßig geformten Erdoberfläche wird durch ihre Koordinaten auf einem Rotationsellipsoid angegeben, das als einfaches Modell der Äquipotentialfläche der Erde (der mittleren Meereshöhe) dient. Das Kartendatum gibt die Achslängen dieses Ellipsoids und seinen Ursprung relativ zum Erdmittelpunkt an. Das Datum WGS84 (World Geodetic System 1984) nähert sich global am besten dieser Äquipotentialfläche (dem Geoid) an. Die Erde ist nicht wirklich eine Kugel sondern am Äquator abgeplattet. In Tabellenbüchern kann man lesen, dass - vom "Erdmittelpunkt" aus gesehen der Pol eine Entfernung von 6356.77 km hat (Polradius), während die Strecke Mittelpunkt - Äquator 6378.16 km misst (Äquatorradius), also wirklich größer ist.

Das Ellipsoid wird durch die Angabe der großen Halbachse a (äquatorialer Radius) und der Abplattung f definiert. Dabei ist $f=(a-b)/a$ und b die kleine Halbachse (vom Erdmittelpunkt zum Pol.) Aus praktischen Gründen werden die Ellipsoide meist relativ zu WGS84 angegeben.

Ellipsoide sind Modelle für die ziemlich unregelmäßig geformte Erdoberfläche, sie entstehen wenn man die ebene Figur Ellipse um ihre kleine Achse rotieren lässt. Es entsteht ein (räumliches) Rotationsellipsoid.

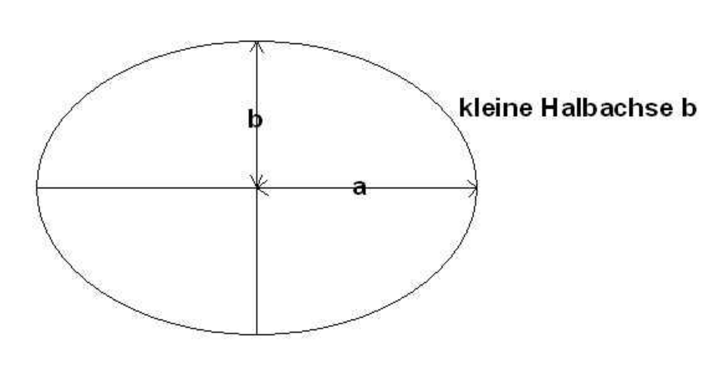


Abbildung 3 Ellipsen-Darstellung

Es lassen sich zwei Gruppen von Koordinatenangaben unterscheiden. Die Koordinaten der Gruppe "Geografisch" zeigen Koordinatenwerte in Grad, d.h. die Positionen werden als geografische (ellipsoidische) Länge und Breite angezeigt, ggf. ergänzt um sog. ellipsoidische Höhen. Es handelt sich also um ellipsoidische Koordinaten, die sich aber hinsichtlich der Art des zugrunde liegenden Ellipsoids unterscheiden. Unsere Beispiele beziehen sich der Reihe nach auf das WGS84-Ellipsoid, das Hayford-Ellipsoid (im Datum ED50), das GRS80-Ellipsoid (im Datum ETRS89) und das Bessel-Ellipsoid (im Potsdam-Datum). Das GRS80- und das WGS84-Ellipsoid stimmen hinsichtlich der Ergebnisse praktisch überein. In der zweiten Gruppe liegen Koordinatenwerte in der Einheit Meter vor und beziehen sich wieder auf verschiedene Ellipsoide: Die Gauß-Krüger-Koordinaten (Bessel-Ellipsoid), und die beiden UTM-Systeme: UTM-ED50 (Bezugsfläche Hayford-Ellipsoid) und UTM-WGS84 (Bezugsfläche WGS84-Ellipsoid).

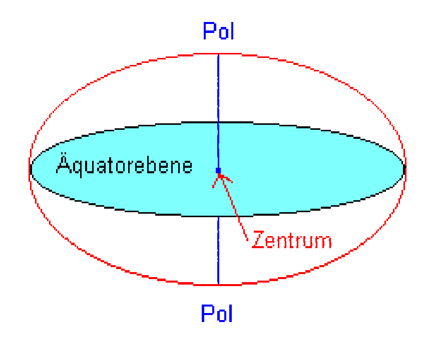


Abbildung 4 Ellipsoidendarstellung

Ellipsoid	***WGS84***	***GRS80***	***Hayford***	***Bessel***
Große Halbachse (a)	6378137.00000 m	6378137.00000 m	6378388.00000 m	6377397.15508 m
Kleine Halbachse (b)	6356752.31425 m	6356752.31414 m	6356752.94613 m	6356078.96290 m

Wie man sieht, sind die Werte alle verschieden aber keines dieser Ellipsoide zeigt genau die Halbachsenwerte. Fast identisch sind die Werte des WGS84- und des GRS80-Ellipsoids.

Die beiden Ellipsoide WGS84 und GRS80 werden mir ihrem Zentrum im "Mittelpunkt der Erde" (geozentrisch) gelagert und ihre kleine Achse in die Polachse ("Erdachse", Gerade vom Nord- zum Südpol) gelegt. Maßgabe der Brauchbarkeit scheint eine möglichst gute "Anschmiegung" an die unregelmäßigere Erdoberfläche zu sein und es ist klar, dass die Beurteilung der "Güte" eines solchen Ellipsoids erst mit "überlegenen" Messmitteln möglich sein wird. Diese hohe Präzision ist vielleicht erst mit Hilfe von Satelliten (wie z.B. GPS-Satelliten) möglich geworden. Die Abweichungen zwischen Erdoberfläche

(eigentlich Geoidfläche) und diesem "mittleren Erdellipsoid" bewegen sich in der Größenordnung von ± 100 m. Die übrigen Ellipsoide haben nicht nur etwas andere Achsenlängen sondern sind, wie man lesen kann, zudem so gelagert, dass sie sich eher für ein begrenzteres Gebiet an die dortige Erdoberfläche (eigentlich Geoidfläche) optimal anschmiegen. Dies bedeutet dann, dass sie mit ihrem Zentrum nicht unbedingt genau ins Erdzentrum gelegt sind und die kleine Achse muss auch nicht notwendigerweise parallel zur Erdachse liegen ("konventionelle Ellipsoide").

Landkarten entstehen aus der dreidimensionalen Vermessung (durch Triangulation) der Erdoberfläche. Die Höhen werden dabei relativ zu einem willkürlich festgelegten Höhenbezugspunkt angegeben. In Deutschland liegt dieser Punkt in der Nähe von Berlin. Von diesem Referenzpunkt aus ist Deutschland mit einem Netz geodätischer Punkte überzogen. Jeder triangulierte Punkt wird auf eine Bezugsfläche projiziert, das Bessel-Geoid (das in seiner Höhe durch den Höhenbezugspunkt gelegt wird). Als Gitternetzkoordinaten werden in Deutschland die Gauß-Krüger Koordinaten benutzt. Sie geben den Abstand in Metern vom Äquator (Hochwert) und einem Referenzmeridian (Rechtswert) an. Die Punkte auf dem Geoid werden dann, z.B. orthographisch, auf eine Kartenebene projiziert. Da Straßenkarten und Stadtpläne richtige Entfernungen angeben sollten, verwendet man hier spezielle Projektionsmethoden.

3.1.5.2. Karten-Projektionen

(Quelle [STU01])

Je nach Verwendungszweck der Karte wird die Erdoberfläche nach unterschiedlichen Algorithmen auf eine ebene (Karten-) Fläche projiziert. Für übliche Landkarten verwendet man die orthographische Projektion, für Seekarten die Mercatorprojektion und für Wetterkarten die gnomonische Projektion.

3.1.5.2.1. Die orthografische Projektion

Bei der orthographischen Projektion wird die Kugeloberfläche wie eine Apfelsinenschale in Spalten abgeschält und eingeebnet. Diese Projektion ist flächentreu. Für Strassenkarten muss man aber eine Korrektur anbringen, um Entfernungen richtig abzubilden. In orthografischen Karten sind die Breitengrade gerade Linien, die Meridiane sind gebogen.

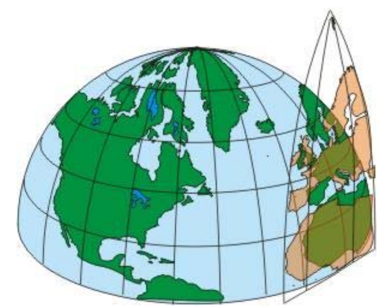


Abbildung 5 orthografische Projektion

3.1.5.2.2. Die Mercator-Projektion

Der Vorteil der von Mercator in der Seefahrt eingeführten Zylinderprojektion - gegenüber der für Landkarten üblichen orthographischen Projektion - ist, dass die Verbindungslinie zwischen zwei Orten direkt den zu steuernden Kurs ablesen lässt: sie ist winkeltreu, bildet aber Flächen nicht richtig ab. Meridiane und Längengerade sind gerade Linien. Der Nachteil liegt in der mit steigender geographischer Höhe wachsenden Verzerrung von waagrechten Strecken.

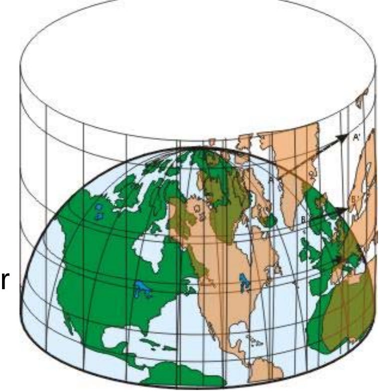


Abbildung 6 Mercator Projektion

3.1.5.2.3. Die gnomonische Projektion

Bei der gnomonischen Projektion wird die Kugeloberfläche auf eine eben Tangentialfläche projiziert. Dabei kann man den Erdmittelpunkt (Wetterkarten, Seekarten) oder den Gegenpol als Ausgangspunkt des Projektionsstrahls verwenden. Die Längengrade sind gerade Linien, die Meridiane Ellipsen bzw. Hyperbeln mit ungleichem Abstand, wenn der Erdmittelpunkt verwendet wird, bzw. gleichem Abstand, wenn der Gegenpol der Anfangspunkt des Projektionsstrahls ist. Zugbahnen von Tiefdruckgebieten (auf Großkreiskurs) sind gerade Linien.



Abbildung 7 Gnomonische Projektion

3.1.6. Nautische Navigation

Die folgenden Angaben und Begriffe sind aus dem Handbuch der Navigation entnommen (Quelle [BÖH01])

3.1.6.1. Navigation

Mit dem Begriff Navigation ist jede Maßnahme (Beobachtung, Messung und Auswertungsmethode), mit welcher der geographische Ort und/oder die Bewegung eines Fahrzeuges ermittelt bzw. ein bestimmtes Fahrtziel, ggf. auf einem festgelegten Weg, erreicht wird.

3.1.6.2. Seemeilen

Entgegen dem normalen Berechnen von Entfernungen in Kilometer werden auf dem Wasser die Entfernungen in Seemeilen [sm] sogenannten nautischen Meilen gemessen. Eine Seemeile entspricht 1852 Meter (Bogenminute eines Großkreises, 360 * 60' = 21600' = 21600 sm, auf der Erdkugel). Der mittlere Großkreisumfang der Erde beträgt 40003,3 km.

3.1.6.3. Knoten

Die Geschwindigkeit von Fahrzeugen auf dem Wasser werden in Knoten ermittelt. Knoten werden in Seemeilen pro Stunde gemessen. Die Anzahl an Seemeilen entspricht der Anzahl an Meridianterien je Sekunde, sodass 1 kn rund 0,5 Meter pro Sekunde [m/s] ergibt. Beim früher verwendeten Handlog wurde die Fahrt anhand des Auslaufens einer mit Knoten gemarkten Leine in einer bestimmten Zeitspanne festgestellt (daher „Knoten“ als

Geschwindigkeitsangabe).

3.1.6.4. Merdiantertie

Eine Merdiantertie ist der 3600ste Teil einer Seemeile bzw. Meridianbogenminute (0,514 m).

3.1.6.5. Fahrt über Grund (FüG)

Die unter Berücksichtigung von Strom ermittelte Fahrt eines Schiffes über dem Meeresgrund. Ein Schiff legt dementsprechend diese Distanz (DüG) über dem Grund zurück.

3.1.6.6. Distanz über Grund (DüG)

Eine Strecke gemessen von Punkt A nach Punkt B bezogen auf den Meeresgrund.

3.1.6.7. Kurs über Grund (KüG)

Mit dem Begriff Kurs über Grund ist der Winkel zwischen rechtsweisend Nord und der Richtung des Weges über Grund beschrieben.

3.1.6.8. Bordzeit (BZ)

Die Bordzeit ist die nach der jeweiligen Zeitzone an Bord geltende Uhrzeit.

3.1.6.9. Loxodrome

Unter einer Loxodrome auf einer Kugel versteht man eine Kurve "festen Kurses", d.h. eine Kurve, die alle Längenkreise unter demselben Kurswinkel schneidet. Jedes Schiff und jedes Flugzeug, welches eine konstante Himmelsrichtung ansteuert, bewegt sich auf einer räumlichen Spirale (loxodrom) um einen der Pole. Die Loxodrome hat somit eine entscheidende Bedeutung in der See- und Luftfahrt wenn es darum geht auf der Erdoberfläche zu navigieren.

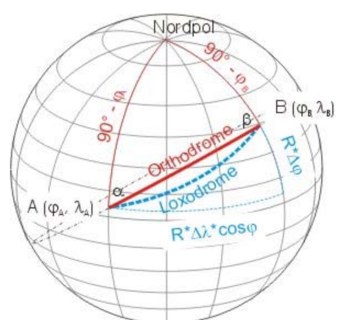


Abbildung 8 Orthodrome und Loxodrome

3.1.6.10. Orthodrome

Die Orthodrome ist die kürzeste Verbindung zweier Punkte auf einer Kugeloberfläche. Die Orthodrome ist immer ein Teilstück eines Großkreises. In der Luftfahrt fliegt man meist entlang dieser Orthodrome, um die geringste Flugstrecke zurücklegen zu müssen.

3.2. Berechnungen

Unabhängig von den im Quelltext gezeigten Berechnungen (die im Kapitel *Systementwurf* beschrieben werden), werden hier die allgemeinen Kenntnisse der verschiedenen Berechnungen der Terrestrischen Besteckrechnung beschrieben.

3.2.1. Nautische Berechnungsverfahren und Terrestrische Besteckrechnung

(Quelle [BÖH01])

3.2.1.1. Mittelbreitenverfahren

Das Mittelbreitenverfahren dient der Ermittlung von Distanzen zwischen zwei Orten A und B oder auch zum Koppeln von einem Ort A mit bekanntem Kurs und bekannter zurückgelegter Distanz. Das Verfahren ist bei niedrigen und mittleren Breiten bei loxodromischen Distanzen bis ca. 600 sm ausreichend genau, wenn der Breitenunterschied dem Betrage nach kleiner als 5° bleibt. Andernfalls ist die Berechnung durch Grosskreisnavigation vorzuziehen. Bei der Besteckrechnung werden die griechischen Symbole λ (Längengrad) und φ (Breitengrad) verwendet.

3.2.1.1.1. Berechnung von Kurs und Distanz aus Punkt A und B

Formeln ($\acute{\alpha}$ als viertelkreisiger Kurswinkel; φ, λ und KüG in °; DüG in sm; Eingabe φ_N bzw. λ_E positiv und φ_S bzw. λ_W negativ; bei einer Übersegelung der Datumsgrenze anstelle von $\lambda_A - \lambda_B$ direkte Eingabe von $\Delta\lambda$; im Fall einer Segelung von der West- zur Ostlänge mit positivem un von der Ost- zur Westlänge mit negativem Vorzeichen)

- **Kurswinkels von Punkt A nach Punkt B**

Berechnung des Kurs über Grund (KüG):

$$\acute{\alpha} = \arctan \{ (\lambda_A - \lambda_B) * \cos [(\varphi_A + \varphi_B) : 2] : (\varphi_A - \varphi_B) \}$$

Entsprechend dem Vorzeichen des viertelkreisigen Kurswinkels $\acute{\alpha}$ gilt für dessen vollkreisige Umwandlung als algebraische Addition bei einer richtungsbezogenen Segelung in den nordöstlichen (NE) Quadranten + $\acute{\alpha}$= KüG, in den südöstlichen (SE) Quadranten – $\acute{\alpha} + 180°$= KüG, in den südwestlichen (SW) Quadranten + $\acute{\alpha} + 180°$= KüG und in den nordwestlichen (NW) Quadranten – $\acute{\alpha} + 360°$= KüG.

- **Distanz von Punkt A nach Punkt B**

Berechnung der Distanz über Grund (DüG):

$$DüG = (\varphi_A - \varphi_B) * 60 : \cos (KüG)$$

Bei Distanzen, die unter Kursen im Bereich von 90° oder 270° berechnet werden sollen, bietet sich folgende Berechnungsart als die bessere an:

$$DüG = (\lambda_A - \lambda_B) * \cos [(\varphi_A + \varphi_B) : 2] * 60 : \sin (KüG)$$

Ein negatives Vorzeichen des Ergebnisses der DüG ist unerheblich

3.2.1.1.2. Berechnung des Bestimmungsort aus Punkt A, Kurs und Distanz

Berechnung des Bestimmungsortes aus dem Abfahrtsort, dem Kurswinkel und der Distanz heraus, Formeln (φ, λ und KüG in °; DüG in sm; Eingabe φ_N bzw. λ_E

positiv und φ_S bzw. λ_W negativ):

$\varphi_B = \varphi_A + DüG * \cos (KüG) : 60$

$\lambda_B = \lambda_A + DüG * \sin (KüG) : \cos [(\varphi_A + \varphi_B) : 2] : 60$

Zum Ergebnis gilt: Bei einem Vorzeichenwechsel (Übersegelung des Äquators oder des Nullmeridian) ändert sich die Benennung; wenn $\lambda_B > 180°$ ist (bei einer Übersegelung der Datumsgrenze), ändert sich die Benennung ebenfalls, wobei dann der Differenzwert zu 360° den Zahlenwert von λ_B ergibt.

3.2.1.2. Grosskreisnavigation

Die Berechnung von Distanz oder Kurswinkel per Grosskreisnavigation, insbesondere auf hohen Breiten und sehr großen Distanzen. Zur Segelung einer Grosskreisdistanz (d_{GK}) als kürzester Verbindung zwischen zwei Orten auf der Erdoberfläche können in der Mercatorkarte Wegepunkte eingetragen werden, an denen der Kurs jeweils als nuerer Grosskreisanfangskurs (AK) nach Berechnung einer bestimmten Distanz um 001° zu ändern ist, um von einem Wegepunkt (Abfahrtsort) zum nächsten Wegepunkt (Bestimmungsort) eine Loxodrome zu segeln.

Falls man nach einer genauen Schiffsortbestimmung eine Versetzung von der beabsichtigten Grosskreissegelung feststellt, muss ein neuer Grosskreisanfangskurs berechnet werden; es wäre falsch, auf den ursprünglichen Grosskreis zurückzukehren, da man dann einen Schlangenlinie um diesen segeln würde, sodass sich der Distanzgewinn verringern würde. Gegebenfalls muss zur Meidung gefährlicher Gewässer eine Grosskreissegelung und eine größere Loxodromsegelung stattfinden, was als „Mischsegeln“ bezeichnet wird.

- **Grosskreisdistanz oder Orthodrome**

Formeln (d_{GK} in sm; φ und λ in °;Eingabe φ_N bzw. λ_E positiv und φ_S bzw. λ_W negativ):

$d_{GK} = \{\arccos [\sin (\varphi_A) * \sin (\varphi_B) + \cos (\varphi_A) * \cos (\varphi_B) * \cos (\lambda_A - \lambda_B)] \} * 60$

- **Grosskreisanfangskurs**

Formeln (AK und φ in °; d_{GK} in sm; Eingabe φ_N positiv und φ_S negativ):

$AK = \arccos \{ [\sin (\varphi_B) - \sin (\varphi_A) * \cos (d_{GK} : 60)] : [\cos (\varphi_A) * \sin (d_{GK} : 60)] \}$

Der halbkreisige AK ergibt sich vollkreisig bei Kursen in den östlichen Halbkreis direkt und in den westlichen Halbkreisen aus 360° - AK.

- **Kursänderung auf einem Grosskreis um 001°**

Näherungsformel (d in sm; φ_A und AK in °):

$d \approx 60 : \tan (\varphi_A) : \sin (AK)$

4. Pflichtenheft

4.1. Zielbestimmungen und Aufgabenstellung

Ausgehend von der schon kurz umrissenen Aufgabenstellung in Kapitel 2, werden hier nun die einzelnen Kriterien der Vorgabe zur Erstellung dieser Software genauer betrachtet.

4.1.1. Mußkriterien

4.1.1.1. Plattformunabhängigkeit

Die Software GPSMan hat die Maßgabe, das Sie unter mehreren Betriebssystemen nutzbar sein muss. Dies war einer der Gründe warum hier die Programmiersprache Java gewählt wurde. Durch ihre Virtual Machine ist auf vielen Betriebssystemen (Windows®, Macintosh®, Linux, Solaris, usw.) die Benutzbarkeit gewährleistet.

4.1.1.2. Bilddateien als Seekarten

Bei der Verwendung von digitalisierten Seekarten muss GPSMan in der Lage sein so genannte KAP-Dateien (dies sind digitalisierte Seekarten von der Fa. MAPTECH INC.) einlesen und darstellen zu können. Anhand der Dateiformatbeschreibung der KAP-Dateien muss hier ein Einlesealgorithmus implementiert werden, der für jegliche Seekarte im KAP Format funktionsfähig ist.

Zusätzlich müssen aber auch „normale“ Bilddateien eingelesen werden können. Hier fiel die Wahl auf das JPEG-Format gelegt. Somit hat der Benutzer die Chance Seekarten einzuscannen und diese im GPSMan als JPEG Image einzulesen und zu verwenden. Hierzu ist aber der Benutzer auch verpflichtet, diese Seekarten mindesten einmal korrekt manuell zu kalibrieren, den nur mit einer korrekten Kalibrierung können diese Seekarten auch als solche im GPSMan verwendet werden. Diese Fähigkeit der manuellen Kalibrierung mit Hilfe grafischer Benutzerschnittstellen ist somit auch ein Mußkriterium zur Erstellung der Software GPSMan. Ist eine JPEG Seekarte manuell kalibriert worden, so sind diese Kalibrierungsdaten in einer Datei abzuspeichern, so dass bei einem erneuten Aufruf dieser JPEG Seekarte die Kalibrierung nicht wiederholt durchgeführt werden muss.

4.1.1.3. Bewegen innerhalb der Seekarten

Zum Bewegen innerhalb der dargestellten Seekarten muss es dem Benutzer möglich sein die Darstellung der jeweiligen Seekarte zu beeinflussen. Diese Beeinflussung soll in Form von Zoomen der Seekarte möglich sein. Hier müssen entsprechende Menüpunkte und Buttons erstellt werden, die dem Benutzer folgende Möglichkeiten bieten

- vergrößern der Seekarte (Zoom +)
- verkleinern der Seekarte (Zoom -)
- zoomen auf Originalgröße (wiederherstellen der ursprünglichen Ansicht der

Seekarte)

- zum Verschieben der Seekarte (bei Karten die größer sind als die momentane Bildschirmauflösung) muss ein System zum Scrollen der Seekarte vorgesehen werden.

4.1.1.4. Wegpunkte

GPSMan muss in der Lage sein Wegpunkte zu erstellen. Wegpunkte sind Punkte auf der Seekarte mit der Darstellung durch verschiedene Symbole. Als Standort der Wegpunkte gelten nur die geographischen Koordianten. D.h. das nach Erstellung der Wegpunkte sofort die anhand der Kalibrierung die Pixelkoordinaten der Wegpunkte in geographische Koordianten umgewandelt werden müssen.

Des weiteren können Wegpunkte so genannten Routen zugeordnet werden. Routen können mehrere Wegpunkte enthalten und werden auf der Seekarte als Verbindungslinien zwischen den Wegpunkten dargestellt.

Die Parameter der Wegpunkte:

- Koordinate als Breiten- und Längengradangabe
- Name des Wegpunktes
- eine Notiz zum Wegpunkt
- Symbol des Wegpunktes
- zugeordnete Routen

Die Parameter einer Route:

- Name der Route
- eine Notiz zur Route
- Farbe der Route

Restriktionen zu Parameter der Wegpunkte und Routen:

Der Name von Wegpunkten und Routen muss in Großbuchstaben geschrieben werden und darf nur Buchstaben, Ziffern und Leerzeichen enthalten.

Wegpunkte und Routen müssen unabhängig voneinander sein, d.h. jeder Wegpunkt kann einer oder mehreren Routen zugeordnet werden, jede Route kann einen oder mehrere Wegpunkte erhalten.

Diese Zuordnungen müssen durch den Benutzer erfolgen und müssen wie alle Parameter der Wegpunkte und Routen änderbar sein. GPSMan muss die Möglichkeit enthalten diese Routen und Wegpunkt in Dateien abzuspeichern und

wieder zu laden zu können. Durch die Tatsache der geographischen Koordianten ist das Laden und Darstellen von Wegpunkten und Routen unabhängig von der jeweiligen Seekarte. Sind die Wegpunkte und Routen im Sichtbereich der Seekarte, so müssen Sie angezeigt werden.

4.1.1.5. Simulation

Ein Simulationsmodus von Schiffsbewegungen ist ein weiteres Mußkriterium dieser Software. Die Simulation bezieht sich auf die Berechnung der Fahrt eines virtuellen Schiffes unter Berücksichtigung möglicher Abweichungen durch Gegenwind oder Gegenströmungen.

Folgende Parameter sind für die Simulation notwendig:

- geographische Position durch Angabe von Breiten- und Längengrad
- Geschwindigkeit des Schiffes in Knoten
- Kurs des Schiffes
- Abweichungsgeschwindigkeit durch Gegenwind oder Gegenströmung in Knoten
- Abweichungswinkel
- Zeitintervall, in dem diese Simulationsberechnungen durchgeführt werden sollen in Sekunden

Die Berechnung der Simulationwerte muss durch die nautische Mittelbreitenberechnung erfolgen.

Für den Benutzer muss es deutlich erkennbar sein, das es sich hier um die Simulation handelt.

4.1.1.6. Trackpunkte

Ähnlich wie die Wegpunkte, so muss es möglich sein so genannte Trackpunkte zu erstellen. Trackpunkte zeigen den zurückgelegten Weg eines Schiffes durch Zeichnen der einzelnen Trackpunkte auf der Seekarte. Unabhängig von der Tatsache ob die Bewegungsdaten von einem GPS Gerät oder von der Simulation stammen, muss GPSMan in der Lage sein, diese Trackpunkte mit verschiedenen Symbole auf der jeweiligen Seekarte in bestimmten Zeitabständen darzustellen.

Die Parameter der Trackpunkte:

- geographische Position durch Angabe von Breiten- und Längengrad
- Name des Trackpunktes
- eine Notiz zum Trackpunkt

- Symbol des Trackpunktes
- Farbe des Trackpunktes
- Zeitintervall zur Darstellung der Trackpunkte

Trackpunkte werden zu so genannten Tracks zusammen gefasst.

Die Parameter von Tracks:

- Name des Track
- eine Notiz zum Track
- Farbe des Track

Wie bei den Wegpunkten, so muss auch hier die Möglichkeit für den Benutzer vorhanden sein die Tracks und Trackpunkte nachträglich zu bearbeiten. Die Positionsangabe der Trackpunkte werden durch das GPS oder durch die Simulation erzeugt, daher darf keine Veränderung der Positionsangaben möglich sein.

Trackpunkte und Tracks müssen in Dateien abgespeichert und wieder geladen werden können. Durch die Verwendung der geographischen Koordinaten ist das Laden und Darstellen von Trackpunkten und Tracks unabhängig von der jeweiligen Seekarte. Sind die Trackpunkte und Tracks im Sichtbereich der Seekarte, so müssen Sie angezeigt werden.

4.1.1.7. Messen von Entfernungen

Es muss dem Benutzer möglich sein Entfernungen auf der Seekarte zu bestimmen. Diese Messungen sollten durch Messlinien auf der Seekarten eingezeichnet bleiben, bis der Benutzer diese wieder explizit entfernt.

4.1.1.8. Berechnen von Kursen

Es muss dem Benutzer möglich sein nautische Kurswinkel auf der Seekarte zu bestimmen. Diese Winkelangaben sollten durch Kurspfeile auf der Seekarten eingezeichnet bleiben, bis der Benutzer diese wieder explizit entfernt.

4.1.1.9. Darstellen der aktuellen Schiffsposition

4.1.1.9.1. Schiffsposition auf der Seekarte

Die Darstellung der aktuellen Schiffsposition muss durch die Darstellung eines Symbols auf der jeweiligen Seekarte möglich sein. Zur Angabe der Position muss die zusätzliche grafische Angabe des Kurses noch hinzugefügt werden.

4.1.1.9.2. Positionsangaben im Tracking Display

Es muss dem Benutzer möglich sein, die aktuellen Positions- und Kursdaten in einem Dialogfenster ersehen zu können.

Die Parameter der Positions- und Kursdaten:

- derzeitige Breitengrad Position
- derzeitige Längengrad Position
- Geschwindigkeit in Knoten
- nautischer Kurswinkel in Grad
- Datum
- aktuelle Bordzeit

4.1.2. Wunschkriterien

4.1.2.1. Darstellen der aktuellen Mausposition in der Seekarte

Es sollte dem Benutzer möglich sein, jederzeit die Positionsdaten des Mauszeiger in der Software zu ersehen (z.B. durch Angabe der Position der Maus in der Statuszeile). Hierzu sollten die Pixelkoordinaten und die geographischen Koordianten ersichtlich sein.

4.1.2.2. Darstellen der Parameterdaten der KAP Dateien

Sofern eine KAP-Datei als Seekarte verwendet wird, sollte es dem Benutzer möglich sein, die Parameterwerte der KAP-Datei einzusehen. Diese Daten liegen in den KAP-Dateien als ASCII Texte vor und sollten in einem Dialogfenster dargestellt werden.

4.1.3. Wunschkriterien außerhalb dieser Diplomarbeit

Aus zeitlichen Gründen sind bei der Erstellung dieser Diplomarbeit nicht alle Wunschkriterien umsetzbar. Ich möchte jedoch diese angesprochenen Kriterien hier kurz aufführen.

4.1.3.1. BSB Seekarten

BSB Karten sind die Hauptkarten bei den digitalisierten Seekarten. Sie enthalten außer den Bilddaten normaler Seekarten auch noch Verweise auf andere KAP Seekarten. So ist eine Navigation innerhalb der verschiedenen Seekarten mithilfe dieser BSB Karten möglich.

Eine BSB Seekarte zeigt alle Detailkarten in ihrem dargestellten Bereich durch einen Rahmen an. Mit Anwahl dieses Rahmens mit der Maus wird automatisch die Detailkarte geladen und in der Anwendung gezeigt. Außer den Detailkarten kann man mit Klick an den Außenrand in die übergelagerte Seekarte verzweigen, d.h. die nächst größere vorliegende Seekarte wird geladen und dargestellt.

4.1.3.2. Logbuch

Jeder Schiffsführer muss ein Logbuch führen. In einem Logbuch werden

Eintragungen zu der Fahrt des Schiffes und evtl. Vorkommnissen gemacht.

Die einzelnen Parameterdaten eines Logbuches liegen nicht vor.

Anmerkung:

Als Vorbereitung zur Implementation des Logbuches wurden in der Menüleiste und der Buttonleiste bereits entsprechende Aufrufmöglichkeiten vorgesehen.

4.1.3.3. Anbindung GPS Geräte

Die Anbindung von GPS Geräten und Einlesealgorithmen zum Auswerten und Darstellen der durch das GPS Gerät gelieferten Daten.

Anmerkung:

Als Vorbereitung zur Implementation des GPS Anbindung wurden in der Menüleiste und den Buttonleiste bereits entsprechende Aufrufmöglichkeiten vorgesehen. Als internes Interface existiert die Klasse GPSTrackingAdapter bereits (allerdings ohne Implementierung). Somit ist eine Schnittstelle für spätere Erweiterung hinsichtlich der Anbindung von GPS Geräten vorbereitet.

4.1.3.4. Bewegen innerhalb der Seekarten mit HandCursor

Zur Vereinfachung der Bewegung innerhalb der Seekarten sollte es möglich sein mit einem HandCursor die Karte zu bedienen. Diese Funktionalität ist z.B. aus Programmen der Fa. Adobe bekannt. Man hält mit dem HandCursor die Karte fest und bewegt Sie nun mit den Bewegungen der Maus. Diese Erweiterung zur Funktionalität des Scrollen ist angesichts der Tatsache, das die Software auf einem Schiff auf dem Meer genutzt werden soll sinnvoll. Da durch die Bewegungen des Schiffes eine exakte Führung der Maus an die Scrollbalken nicht immer gewährleistet werden kann und mit der Funktionalität des HandCursor so eine Vereinfachung für den Benutzer erfolgt.

4.1.3.5. Anwählen von Objekten in der Seekarte mit einem Mausklick

Es sollte dem Benutzer außerhalb der verschiedenen Verwaltungsfunktionen von Wegpunkten und Trackpunkten möglich sein, mit einem Mausklick in eine jeweilige Seekarte den bestimmten Weg- oder Trackpunkt bearbeiten zu können.

4.2. Einsatz

4.2.1. Anwendungsbereiche

Der Anwendungsbereich der Software GPSMan ist im Bereich der Navigation auf Segel- und kleinen Motorschiffen zu finden. Mithilfe der visuellen Darstellungsmöglichkeiten der Position des Schiffes wird dem Führer eines Schiffes die Navigation erheblich vereinfacht.

Mit der Simulation der Schiffspositionen können auf angehende Schiffsführer die Handhabung der Navigation per GPS üben.

4.2.2. Zielgruppen

Diese Software ist gedacht für Schiffsführer und angehende Schiffsführer von kleinen bis mittleren Segel- und Motorschiffen.

4.2.3. Betriebsbedingungen

Die Software kann beim Navigieren auf der Fahrt mit Schiffen zur visuellen Unterstützung der Navigation genutzt werden. Mit der Fähigkeit der Simulation kann sie aber auch in allen Bereichen verwendet werden.

4.3. Umgebung

4.3.1. Software

Um eine lauffähige Version der Software GPSMan zu erhalten, ist es zwingend erforderlich, dass die Runtime von Java in der Version 1.4.0 auf dem System korrekt installiert ist. Spätere Versionen (zum Zeitpunkt der Dokumentation lagen die Versionen 1.4.1 und 1.4.2 als neuere Versionen von Java im Internet frei zum Download vor) können ebenfalls zum Einsatz kommen. Die Software Java ist kostenfrei im Internet unter http://java.sun.com für viele Betriebssysteme zu erhalten.

Weitere Bedingungen zur Lauffähigkeit von GPSMan gibt es nicht.

4.3.1.1. Betriebssysteme

Das Projekt wurde unter SUSE Linux 9.0 prof. implementiert und unter Windows XP® prof. und MacOS X 10.2 getestet. Der Einsatz der Software ist unter jedem Betriebssystem möglich, welches Java ab Version 1.4.0 unterstützt.

4.3.2. Hardware

Als Mindestkonfiguration für das Programm GPSMan können alle Rechner verwendet werden, auf denen Java JDK 1.4.0 lauffähig ist. Aufgrund der Grösse mancher Karten ist eine Hauptspeicherausstattung von mindestens 256MB empfehlenswert.

Weitere Beschränkungen aus Sicht der Hardwarekonfiguration gibt es nicht.

4.4. Daten

Die von der Anwendung erzeugten Daten sollten im Format XML abgespeichert werden. Die Entscheidung für XML basiert auf der Tatsache, dass XML Dateien standardisiert sind und mit jedem herkömmlichen Browser eingesehen und evtl. auch bearbeitet werden können.

Für das Abspeichern der Kalibrierungsdateien bei manueller Kalibrierung sollte ein nicht einsehbares binäres Dateiformat gewählt werden.

4.5. Entwicklungsumgebung

Die zur Entwicklung dieser Diplomarbeit genutzten Elemente werden wie folgt beschrieben.

4.5.1. Programmiersprache

Die Softwareelemente, die zur Implementierung der Diplomarbeit notwendig waren, werden wie folgt aufgelistet:

- Java SDK 1.4.0, 1.4.1., 1.4.2 (lauffähig ist die Software GPSMan ab Version 1.4.0)
- Eclipse Java Entwicklungsumgebung Version 2.1.2 von IBM® (OpenSource Software)
    - mit dem Plugin Jalopy Formatierungstool
    - mit dem Plugin XML Darstellung
    - mit dem Plugin CVS zur Sicherung von Softwareständen

Zur Implementierung des Projektes kann jeder herkömmliche Editor genutzt werden, d.h. Eclipse ist nicht zwingend notwendig, jedoch wesentlich komfortabler in der Benutzung.

4.5.2. Hardware

Die bei diesem Projekt zur Entwicklung verwendete Hardware war wie folgt:

- Notebook DELL® Inspirion 8200

  1,6 GHz INTEL® Prozessor, 512 MB RAM, 80 GB HDD, 15'' TFT Display (Auflösung 1600 * 1200 Pixel)

4.5.3. Dokumentation

Die Dokumentation der Diplomarbeit wurde mit OpenOffice.org 1.1.0 erstellt.

4.5.4. Dateien

Folgende Arbeiten und Programme wurden mir vom Projektleiter Prof. Dr. Dipl.Ing. F.N. Rudolph zur Ansitz und Vorbereitung zur Verfügung gestellt.

- Vorentwurf der Software GPSMan von Bernd Lehmann
- die Projektdokumentation BSB Header von Pascal Klaes
- die Projektdokumentation und die Software WeroGPS von Nadine Druckenmüller
- Dateistrukturbeschreibung der KAP Dateien
- Vorentwurf der Software MapFlex von Prof.Dr.Dipl.Ing.F.N. Rudolph
- Berechnungsklasse des Gauß'schen Algorithmus von Prof.Dr.Dipl.Ing.F.N. Rudolph

4.5.5. Bücher

Die aufgeführten Bücher sind im Rahmen der Diplomarbeit genutzt worden.

4.5.5.1. Zur Einarbeitung in Java

- Core Java, Band1 Grundlagen [CORE1]
- Core Java Band 2, Expertenwissen [CORE2]
- Core Java, graphic AWT [COREAWT]
- Core Java, graphic Swing [CORESWING]
- Java Grundlagen [Java-mitp]

4.5.5.2. zur Einarbeitung in die nautische Navigation

- Handbuch der Navigation [BÖH01]

4.5.5.3. Kartenlesen und GPS

- Orientierung mit Kompass und GPS [HÖH01]
- Richtig Kartenlesen [SCHW01]

4.5.5.4. Dokumentation

- Kompendium OpenOffice.org [BOR01]

4.5.6. Internet

Auf den aufgeführten Internetseiten wurden Informationen für die Erstellung dieser Diplomarbeit entnommen, bzw. wurden diese Internetseiten für den Wissenaufbau über die Themen GPS, Informationen zu Seekarten und Navigation genutzt.

- http://www.astrosails.de [AST01]

  Grundsatzwissen über die nautische Navigation.

- http://www.nautictools.de [NAU01]

  Die Freeware NauticTools wurde zur Überprüfung der nautischen Berechnungen genutzt.

- http://www.rainerstumpe.de [STU01]

  Grundsatzwissen über die nautische Navigation.

- http://otmarlabonde.de.de [LAB01]

  Erläuterungen Ellipsoide und Projektionen.

Bei diesen Internetseiten handelt es sich weitgehend um private Initiativen zur Vermittlung von Wissen über das Thema Navigation.

5. Systementwurf und Implementierung

5.1. Modulhierachie und Ihre Spezifikationen

Das Programm GPSMan ist in folgende Pakete (Packages) unterteilt:

Bezeichnung	***Erläuterung***
GPSMan.Calibration	Kalibrierung und Umrechnung von Pixel in Längen- und Breitengradkooridinaten
GPSMan.KAP	Auslesen der Referenzdaten und Bilddaten aus den KAP-Seekartendateien
GPSMan.Measurement	Messungen von Distanzen und Berechnen von Kursen
GPSMan.Tracking	Simulation und Darstellung von GPS Daten
GPSMan.Util	Allgemeine Toolmethoden (wie z.B. Satz des Pythagoras usw.)
GPSMan.Waypoint	Erzeugung und Verwaltung von Wegepunkten und Routen
GPSMan	Allgemeine Klasse für das Softwareprojekt -Application : Startklasse des Projektes -GPSManFrame : Hauptdialog des Projekt -ImagePane : enthält Methode zum Zeichnen -PrefData : Datenklasse der Einstellungen -PrefDialog : Dialog der Einstellungen

Hier nun die genaueren Erläuterungen zu den einzelnen Packages.

Im Rahmen der Beschreibungen des Systementwurfes der Software GPSMan werden aus dem Quellcode keine Implementationen hinsichtlich der grafischen Benutzeroberfläche oder anderen Standardverfahren in Java (Eventhandling, Listener, usw.) erläutert und dargestellt.

5.1.1. Package Calibration

Die Kalibrierung der Seekarten erfolgt in mit verschiedenen Verfahren. Bei manuell eingescannten Seekarten im JPEGDateiformat wird ein über drei eingegebene Punkte berechnetes Kalibrierungsverfahren mit Hilfe des Gauss-Algorithmus angewendet. Die digitalen Seekarten im KAP Dateiformat von der Fa. MAPTECH INC. USA haben allerdings normalerweise mehr als drei Referenzpunkte, so dass hier die Anwendung des Dreipunktverfahren nicht anwendbar wäre. Aus Gründen der Eingrenzung meiner Arbeiten an diesem Projekt ist eine Implementierung des Kalibrieren nach mehreren Punkten (>3) nicht erfolgt. Dieser Punkt wird durch ein anderes Projekt bei der Weiterentwicklung dieser Software erfolgen. Daher erläutere ich ein Verfahren,

nachdem ich unter den „vielen“ Referenzpunkten, die drei besten auswähle um dann wieder das Dreipunkt-Verfahren anzuwenden.

5.1.1.1. Klasse CalDialog

Die Klasse CalDialog beinhaltet das grafische Interface zur Eingabe der manuellen Kalibrierung im Falle des Einsatzes einer Seekarte im JPEG-Format.

5.1.1.2. Klasse Calibration

5.1.1.2.1. Kalibrierungsverfahren

Die Klasse Calibration ist für die Umrechnung der Koordinaten zuständig. In ihr werden Pixelkoordinaten in geographische Koordinaten und geographische Koordinaten in Pixelkoordinaten umgerechnet. Bei der jeweiligen Umrechung wird auch die Drehung der Karte berücksichtigt. Die Umrechnung erfolgt mit Hilfe von Matrizen (ema[] für die Umrechnung von Pixelkoordinaten in geographische Koordinaten, emaRev[] für die Umrechung von geographische Koordinaten in Pixelkoordinaten). Diese Matrizen werden durch Lösen eines linearen Gleichungssystems berechnet, welches die Transformation des geographische Koordinatensystems in das Bildschirmkoordinatensystem für eine Karte beschreibt.

5.1.1.2.2. Laden und Speichern von Kalibrierungsdateien

Desweitern ist die Klasse Calibration in der Lage, einmal gemachte manuelle Kalibrierungen in einer binären Datei abzuspeichern. Als Dateiname wird der Name der Seekarte genommen und um die Typerweiterung *.cal erweitert. So wird aus „test.jpg“ die Konfigurationsdatei „test.jpg.cal“. Kalibrierungsdateien können auch von der Klasse Calibration eingelesen werden, so daß einmal gemachte manuelle Kalibrierungen nicht wiederholt gemacht werden müssen. Bei Kalibrierungsdateien, die im gleichen Verzeichnis wie die Seekartendateien liegen, wird dieser Vorgang automatisch ausgeführt.

5.1.1.2.3. Auswahlverfahren für die Referenzpunkte der KAP-Dateien

Zusätzlich zu der manuellen Kalibrierung werden in der Klasse Calibration die KAP-Dateien automatisch kalibriert. Da, wie bereits beschrieben, für ein korrektes Kalibrieren mit mehr als drei Kalibrierungspunkten das implementierte Berechnungsverfahren nicht vorgesehen ist, werden hier aus der Menge der vorhandenen Referenzpunkte die drei am weitesten voneinander entfernten Referenzpunkte ausgewählt. Mit diesen Referenzpunkten wird dann die Kalibrierung nach dem vorliegenden Verfahren berechnet.

Die Auswahl erfolgt durch Vergleich der einzelnen Referenzpunkten mit den äußersten Eckpunkten der Seekarte. Da in der KAP-Datei die max. Auflösung der Seekarte als Zahlenwert bekannt ist (siehe Kapitel *Fachliches Umfeld / Seekartenbildformate / KAP Format*), ist die Bestimmung der äußersten Eckpunkte trivial

```
Coordinate leftTop = new Coordinate();
Coordinate leftBot = new Coordinate(0, sizeY);
Coordinate rightBot = new Coordinate(sizeX, sizeY);
Coordinate tempCoord = new Coordinate();
```

So werden nun alle Referenzpunkte mit den einzelnen Eckpunkten verglichen.

```
// Suche den Punkt mit dem kürzesten Weg zu leftTop
      temp = Tool.pythagoras (leftTop, rightBot);

      for (int i = 0; i < aData.length; i++)
      {
         tempCoord.setX (aData[i][0]);
         tempCoord.setY (aData[i][1]);

         if (Tool.pythagoras (leftTop, tempCoord) < temp)
         {
            temp = Tool.pythagoras (leftTop, tempCoord);
            bestPoints[0] = i;
         }
      }
```

Nach Bestimmung aller drei Kalibrierungspunkten werden diese in die bestehende Datenklasse CalPoints eingefügt und die Kalibrierung wird automatisch durchgeführt.

```
Coordinate point = new Coordinate();

// Setzen des ersten Kalibrierungspunktes
point.setLocation (aData[bestPoints[0]][0], aData[bestPoints[0]][1]);
calPoints.setCalPointImg (point, 0);
point.setLocation (aData[bestPoints[0]][3], aData[bestPoints[0]][2]);
calPoints.setCalPointGeo (point, 0);
```

Diese Form der Kalibrierung kann nicht als endgültige Lösung für den Sachverhalt der Kalibrierung von KAP-Dateien mit mehr als drei Referenzpunkten gelten. Im Rahmen dieser Diplomarbeit standen allerdings nicht genügende Ressourcen zeitlicher Natur zur Verfügung. In Abstimmung mit dem Projektleiter wurde vorerst diese pragmatische Lösung gewählt.

5.1.1.3. Klasse CalPoints

Die Klasse CalPoints ist die Datenklasse der Kalibrierungspunkte. Kalibrierungspunkte bestehen aus zwei Koordinaten, die jeweils die Pixelposition und die geographische Position (Breiten- und Längengrad) darstellen.

```
private Coordinate pointsImg[] = new Coordinate[3];
private Coordinate pointsGeo[] = new Coordinate[3];
```

5.1.1.4. Klasse LinGls

Die Klasse LinGls enthält Standardalgorithmen der linearen Algebra.

5.1.2. Package KAP

Im Package KAP werden alle Klassen gesammelt, die mit der Auswertung und Decodierung der KAP Dateien in Zusammenhang gebracht werden können. Die KAP-Dateien sind digitalisierte Seekarten der Firma MAPTECH INC. In diesen Dateien werden eingescannte Seekarten in einem speziellen Format hinterlegt,

so dass diese Dateien sehr effizient und klein in ihrer Dateigröße sind. Aufgrund der Tatsache, dass Seekarten selten über viele Farben (meist zwischen 4 und 16 verschiedenen Farben) verfügen, werden hier mit dem NOS Kompressionsverfahren und der indizierten Farbtabelle erhebliche Kompressionserfolge erzielt.

Hier nun die einzelnen Klassen mit Ihren Aufgabenstellungen:

Klasse	***Aufgabenstellung***
KapInfo	Auswertung des ASCII Teils der KAP-Datei
KapInfoFrame	Darstellung der Parameter der KAP-Datei
KapConverter	Decodierung der binären Bilddaten in der KAP-Datei

5.1.2.1. Klasse KapInfo

In der Klasse KapInfo werden der ASCII Datenteil der KAP-Kartendatei ausgewertet und dementsprechend Schnittstellen für andere Objekte zur Verfügung gestellt.

Vorab jedoch erst ein Beispiel zur Verdeutlichung der vorhandenen Daten (Auszug aus der Datei world.kap):

```
REF/93,7720,613,70.0000000000,165.0000000000

REF/94,7719,900,65.0000000000,165.0000000000

PLY/1,-59.9783166000,-179.9984666000

PLY/2,75.0074166000,-179.6816833000

PLY/3,75.0031000000,179.9964666000

PLY/4,-59.9925666000,179.9556166000

DTM/0,0

CPH/0

WPX/3,4.1324143780587019e+003,2.1739688611878339e+001,-4.1904016425821311e-002

    4.2568668927100983e-005,3.8703904920662125e-005,4.9643470794369371e-004

    2.1884951179080142e-007,-3.0190747949315798e-007,-8.1453903038549816e-007

    2.3717934215950615e-006

WPY/5,2.7786072675652649e+003,-1.9540522998668399e-002,-2.2144745084944540e+001

    1.2001013840266805e-005,-4.3520314774853505e-005,3.8500292191741003e-003

    2.2293513021452772e-007,6.9615589327377565e-007,1.7384988575722245e-006

    -5.2085607984558163e-004,-8.9291961176093167e-007,-2.6445074146069582e-007
```

Die einzelnen Kennungen sind hier rot markiert, die Nummerierungen der Kennungen blau (es werden nur die Anzahl der Kennungen manuell ermittelt). Alles andere sind die einzelnen Werte der Kennungen.

Zum Einlese Algorithmus muss man konzeptionell folgendes sagen:

- Man muss erst in Erfahrung bringen, welche Kennungen überhaupt vorhanden sind und welche nicht, da es hier keine festes Format der KAP-Dateien gibt. Die vorliegende Anzahl und Art der Kennungen ist meist unterschiedlich.

- Dann muss man die Anzahl der Kennungen einlesen (siehe im Beispiel 4 x PLY)

- Dann benötigt man die Anzahl der einzelnen Werte der Kennungen (siehe Kennung REF, hier sind 4 Werte pro Kennung vorhanden).

Mit diesen Daten kann man den Einlese und Umwandlungsvorgang einleiten (einige Werte liegen als Zeichenkette vor, müssen aber als numerische Werte zur Verfügung gestellt werden).

Hier nun die Methodik zum Einlesen der einzelnen Kennungen:

Auszug aus der Methode setData()

```
if (ver != -1)
    {
      versionOK = true;
    }

    copyright = buffer.indexOf ("!Copyright");

    if (copyright != -1)
    {
      ownerOK = true;
    }

    if (ownerOK)
    {
      this.setOwnerName (buffer.substring (copyright, ver));
    }

    if (versionOK)
    {
      this.setVersionName (buffer.substring (ver, bsb));
    }

    this.setBSBInfo (buffer.substring (bsb, knp));
    this.setKNPInfo (buffer.substring (knp, ced));
    this.setCEDInfo (buffer.substring (ced, ost));

    if (OST_OK)
    {
      this.setOST (buffer.substring (ost, ifm));
    }
```

```
if (IFM_OK)
{
   this.setIFMInfo (buffer.substring (ifm, rgb));
}

if (RGB_OK)
{
   this.setRGB (buffer.substring (rgb, ref));
}

if (REF_OK)
{
   this.setREF (buffer.substring (ref, ply));
}

if (PLY_OK)
{
   this.setPLY (buffer.substring (ply, dtm));
}

if (DTM_OK)
{
   this.setDTM (buffer.substring (dtm, cph));
}

if (CPH_OK)
{
   this.setCPH (buffer.substring (cph, wpx));
}

if (WPX_OK)
{
   this.setWPX (buffer.substring (wpx, wpy));
}

if (WPY_OK)
{
   this.setWPY (buffer.substring (wpy, pwx));
}

if (PWX_OK)
{
   this.setPWX (buffer.substring (pwx, pwy));
}

if (PWY_OK)
{
   this.setPWY (buffer.substring (pwy, err));
}

if (ERR_OK)
{
   this.setERR (buffer.substring (err));
}
```

Der gesamte ASCII Datenteil wird als Zeichenkette in eine Variable eingelesen. Diese große Ursprungszeichenkette wird aufgeteilt in die einzelnen Ursprungszeichenketten der einzelnen Kennungen (siehe Auszug aus der Methode setData() der Klasse KapInfo). Diese aufgeteilten Zeichenketten werden dann an die entsprechenden set Methoden der einzelnen Kennungen zwecks Aufschlüsselung der Werte übergeben. Mit Hilfe der Hilfsklasse Tool

werden die Daten aus den großen Zeichenketten extrahiert und in get Methoden zur Anzeige in dem Objekt der Klasse KapInfoFrame und als get Methode für eine spätere Verwendung zur Verfügung gestellt.

5.1.2.2. Klasse KapInfoFrame

Die im Objekt der Klasse KapInfo ermittelten Parameterwerte werden hier in einem Dialogfenster dargestellt. Hier können die Werte nun vom Benutzer eingesehen werden und auf andere Art und Weise genutzt werden.

5.1.2.3. Klasse KapConverter

Die Klasse KapConverter ist zur Decodierung der binären Bilddaten geschaffen worden. In ihr werden der binäre Teil der KAP Datei aus der jeweiligen NOS Kompression entschlüsselt und in unkodierter Form zur Anzeige und Darstellung an ein Objekt der Klasse ImageIcon weitergegeben.

Hierzu wird nun ausführlich die Methodik zum Decodieren beschrieben. Zuzüglich werden auch allgemeine Informationen zu dem Thema Kodierung (hier speziell NOS Kompression, RLE Run Length Encoding, usw.) hinzugefügt.

5.1.2.4. NOS Kompressionsverfahren

5.1.2.4.1. Farbkodierung oder indizierte Farben

Zu jedem Bildpixel werden NICHT die tatsächlichen Farbwerte hinterlegt, sondern Indizies die auf eine Farbtabelle verweisen.

Beispiel: hier Farbtabelle mit 4 Farben

Index	*Farbwerte RGB*
1	0/0/0
2	255/255/255
3	125/85/235
4	10/100/185

D.h. bei jedem Bildpixel ist nur die Informationen des Farbindex vorhanden. Die Farbtabelle ist hier im ASCII Teil der Datei hinterlegt und kann mittels der Methode getRGB() der Klasse KapInfo übertragen werden. Aufgrund der maximalen Kodierung von sieben Bit kann die Tabelle von mindesten 2 bis maximal 128 Farben enthalten.

5.1.2.4.2. Run Length Encoding (RLE)

Das ''Run Length Encoding'' ist eine verlustfreie, symmetrische, nicht adaptive, physikalische Kompressionsmethode. Sie ist nur vorteilhaft, wenn der Datenstrom längere Folgen von identischen Daten hat. Dies ist bei den Seekarten sehr oft der Fall da in einer Bildzeile viele Pixel gleicher Farbe neben einander liegen. Die Seekarten haben meist auch relativ wenig verschiedene Farben (<10), so dass ein Kodieren mit RLE von Vorteil ist. Im einfachsten Fall ist dies eine flächige Computerzeichnung (hier unsere Seekarten), in der viele benachbarte Pixel die selbe Farbe haben. Beim ''Run Length Encoding'' wird nun

ein Kennzeichen vereinbart, das anzeigt, dass das nächste Datenwort kein Pixelwert ist, sondern die Anzahl angibt, wie häufig der übernächste Wert wiederholt wird. Sollte das Kennzeichen auch im normalen Datenstrom als normaler Datenwert vorkommen können, so muss an dieser Stelle eine Einfachwiederholung in Kauf genommen werden.

Die Folge

1234111422227855555788888888896666666

würde zu

123403140427805570989066

wobei die ''0'' das Wiederholungskennzeichen ist. (Blau zeigt die Anzahl oder auch Lauflänge, Rot den Farbindex der Bildpixel)

Dieses Verfahren findet Anwendung bei TIFF-Dateien, beim ''PC Paintbrush File Format (.pcx)'' , beim GIF Format oder beim ''Utah Raster File Format (.rle). Das Kodierverfahren wurde im Falle der NOS Kompression nicht byteweise sondern variabel bitweise verwendet. D.h. es können hier zwischen einem und sieben Bit für die Farbinformationen benutzt werden. Diesen Wert entnimmt man dem ASCII Teil der Datei, bzw. die Methode getIFM() der Klasse KapInfo liefert diesen Wert. Der Rest des Datenbyte wird dann für die Kodierung der Lauflänge genutzt. Wenn der Wertebereich der restlichen Bit's in einem Byte nicht mehr für die Darstellung der Lauflänge der Bildpixel ausreicht, wird hier mit dem achten Bit eine Weiterverkettung gekennzeichnet. Dies besagt, dass das nächste Byte die restlichen Lauflängeninformationen enthält. Dies kann sich auch über mehrere Byte erstrecken, so das durch die Weiterverkettung beliebig grosse Werte darstellbar sind. Daher ist in jedem Byte das 8. Bit für die Kennzeichnung dieser Weiterverkettung reserviert. Im Algorithmus erreichbar man dies in dem man den bisherigen dekodierten Lauflängenwert (im ersten Byte) mit 128 multipliziert (binäres Verschieben um 7 Bit) und die niederwertigsten 7 Bit des nächsten (verketteten) Byte addiert. Dieser Vorgang ist zu wiederholen, bis keine Kennzeichnung einer Weiterverkettung mehr vorhanden ist.

5.1.2.4.3. Methode convertKapFile()

Als eine der Kernmethoden des Projektes erläutere ich hier zunächst den Quellcode. Später folgen dann die fehlenden kausalen Zusammenhänge der einzelnen Programmelemente:

```
kapFile = new FileInputStream(kapInfo.getPathName ());
```

Hier wird ein Stream zum Einlesen der Daten erzeugt.

```
int bitsPerPixel = kapInfo.getIFM ();
```

Aus dem ASCII Teil wird hier die vorliegende NOS Kodierung mittels der Methode getIFM() ausgelesen. Hieraus resultiert später die Tatsache wieviele Bits pro Byte für die Farbkodierung verwendet werden.

```
int tempLine[] = new int[pixelX];
```

tempLine ist ein temporärer Behälter für die Daten eine Bilddatenzeile

```
// Bis zum 1. Null-Byte lesen (Trenner zwischen ASCII und BinÃ¤r-Teil)
    while (getNextByte () != 0)
    {
      // leere Schleife
    }
```

Da in der KAP-Datei erst der ASCII Teil beschrieben ist, muss sich der Algorithmus erstmal bis zur Kennzeichnung der Bilddatensektion einlesen, die durch ein Nullbyte gesetzt ist. Erst dann können die wahren Bilddaten eingelesen werden. Die while Schleife ist leer, da hier keine weiteren Aktionen ausgeführt werden.

```
// 1. Byte des Binär-Teils überspringen (Kompressionskennung)
getNextByte ();
```

Das erste Byte der Bilddatensektion ist die Kennzeichnung der vorliegende NOS Kompression. Da diese Informationen ja schon durch den ASCII Teil ermittelt wurde, kann man hier dieses Byte überspringen (es wird eigentlich eingelesen, aber nicht genutzt).

```
while (true)
{
        int lineNum;
```

lineNum ist die aktuelle Zeilennummer der Bilddatenzeile

```
        byte temp;
```

temp ist eine temporäre Variable für ein Byte der Bilddaten

```
        int curX;
```

curX ist die Startpositionsangabe für den nächsten Pixelblock der Bilddaten.

```
                // --- Zeilennummer decodieren ---
                temp = getNextByte ();
```

das nächste Byte wird in die temporäre Variable temp übergeben.

```
                // In mindestens einer KAP-Datei treten an dieser Stelle
                // zusätzliche Nullbytes auf (nicht dokumentiertes "Feature" oder
                // Datenmüll), die übersprungen werden müssen.
                while (temp == 0)
                {
                        temp = getNextByte ();
                }
```

In mindestens einer KAP-Datei treten am Zeilenende zusätzliche Nullbytes auf (nicht dokumentiertes "Feature" oder Datenmüll), die übersprungen werden müssen.

```
lineNum = (int) (temp & 0x7F);

while ((temp & 0x80) != 0)
{
        temp = getNextByte ();
        lineNum *= 128;
        lineNum += (int) (temp & 0x7F);
}
```

der Variable lineNum wird der entsprechende Wert, in diesem Fall die Zeilennummer zugewiesen. Aufgrund der Bitmaske 0x7F wird das 8. Bit wegmaskiert und der Wert der Zeilennummer ausgelesen. Auch der Fall der Weiterverkettung ist hier berücksichtigt (Zeilennummern > 127).

```
curX = 0;
```

Intialisiere die Startposition für eine neue Bilddatenzeile.

```
// --- Bilddaten für aktuelle Zeile decodieren ---

while (true)
{
        temp = getNextByte ();

        if (temp == 0)
        {
                if (curX < pixelX)
                {
                        ++curX;
                        continue;
                }
                else
                {
                        break;
                }
        }

int colorIndex = (int) (temp & masks[bitsPerPixel - 1][0]) >>> (7
-bitsPerPixel);
```

Hier wird die Farbinformation des aktuellen Bildpixelblocks ausgelesen und der Variablen colorIndex zugewiesen. Die dem Kompressionverfahren entsprechende Bitmaske wird benutzt um die Farbinformation zu extrahieren. Da diese Information „mitten“ im Byte stehen kann, muss hier eine Verschiebung der binären Information erfolgen, so dass wir einen tatsächlichen Farbindex erhalten.

Beispiel:

00110100 => 00001101 (bitweises Verschieben)

Des weiteren können auch hier mitten in der Bildsektion Nullbytes auftauchen. Diese werden ordnungsgemäss als Pixel mitgezählt, bekommen jedoch keine Farbzuordnung (siehe spätere Anmerkung dazu).

```
int runLength = (int) (temp & masks[bitsPerPixel - 1][1]);

while ((temp & 0x80) != 0)
{
        temp = getNextByte ();
        runLength *= 128;
        runLength += (int) (temp & 0x7F);
}

runLength += 1;
```

Hier wird die Lauflänge des aktuellen Bildpixelblocks ausgelesen und der Variablen runLength zugewiesen. Die dem Kompressionverfahren entsprechende Bitmaske wird benutzt um die Lauflängen Information zu extrahieren. Auch der Fall der Weiterverkettung ist hier berücksichtigt.

```
        for (int j = 0; j < runLength; ++j)
        {
                tempLine[j] = colorIndex - 1;
        }
              }
```

Hier wird die vorliegende RLE Komprimierung wieder dekomprimiert und jedes Pixel erhält hier die Zuordnung seines jeweiligen Farbindexes. D.h. hier werden die unkomprimierten Bilddaten zusammengesetzt.

```
        raster.setPixels (curX, lineNum - 1, runLength, 1, tempLine);
        curX += runLength;
}
```

Aus der temporären Variable tempLine werden die Daten in ein Objekt der Klasse WriteableRaster (hier destRaster) übertragen.

```
if ((lineNum == pixelY) && (curX >= pixelX))

{
        break;
}
```

Nach Überprüfung ob alle Daten der Bilddatensektion eingelesen sind, wird die while Schleife mit break abgeschlossen und verlassen.

```
kapFile.close ();
```

Der InputStream wird geschlossen.

```
kapFile = null;
```

Zusammenhänge:

Die Seekartendatei wird mittels eines Streams geöffnet und ab der Kennung eines Bytes mit Wert = 0x00 (Nullbyte) werden die binären Bilddaten eingelesen.

In einer while Schleife, die beendet wird wenn laut Auflösungwert pixelY und pixelX alle Bilddaten eingelesen sind, werden hier die komprimierten Bilddaten eingelesen und dekomprimiert.

Hierzu nun ein plastisches Beispiel anhand eines Auszuges (Beginn der binären Bildsektion) aus der Datei n0s35_5.kap:

00 04 01 90 89 2C 0B 00 ...

- die Bilddatensektion startet mit einem Nullbyte (grün markiert)
- dann wird die Art der Kompressionsmethode gezeigt (grau markiert). Dieser Wert wird aber schon im ASCII Teil entnommen. Dieser Wert entspricht der Kompressionsmethode Nummer 4 (NOS Kompression Type 4, 4 bit color (8 to 15 colors)).
- Das nächste Zeichen ist die Zeilennummer der Bildzeile (pink markiert). Dies ist in unserem Fall die Bildzeile Nummer 1. Die Berechnung der Zeilennummer geschieht durch die binäre UND Verknüpfung der Bitmaske 0x7F und des vorhandenen Wertes

  0x7F => 01111111

  01 => 00000001

  Hierdurch wird das achte Bit wegmaskiert und nicht bei dieser Berechnung betrachtet. Als Ergebnis der UND Verknüpfung wird der Wert 1 erzielt. Die Zeilennummer beträgt die Nummer 1. Des weiteren wird anschließend überprüft, ob hier keine Weiterverkettung vorliegt (Prüfung ob das achte Bit gesetzt ist). Dies wird bei Zeilennummern die größer werden als dezimal 127 genutzt, da ja die anderen 7 Bit nicht mehr Daten aufnehmen können. In unserem Beispiel ist aber keine Weiterverkettung eingetragen. Der Algorithmus zur Weiterverkettung wird später ausführlich beschrieben.

- die nun folgenden Bytes werden wir später noch genauer untersuchen (rot markiert)
- als letztes aus unserem Beispiel (wieder grün markiert) beginnt eine weitere Bildzeile mit einem Nullbyte

Hier nun die genauere Betrachtung der eigentlichen Bilddaten aus unserem Beispiel:

90 89 2C 0B

- Betrachten wir zunächst den ersten Wert (hexadezimal) 90, der dem binären Wert 10010000 entspricht. In diesem Byte ist der Farbindex der ersten oder des ersten Bildpixels enthalten. Das achte Bit wird auch weg maskiert, da dieses Bit ausschließlich für die Kennzeichnung der Weiterverkettung reserviert ist (rot markiert). Wir erhalten somit folgenden binären Wert 0010000. Nun wird der Farbindex der NOS Kompression aus maskiert und ermittelt. Wie bereits im ASCII Teil ausgelesen und auch hier im Beispiel am Beginn der Bilddatensektion gesehen handelt es sich um die 4 Bit Kodierung. D.h. nur die ersten 4 Bit sind hier die vorliegende Information bzgl. des Farbindexes (grün markiert). Daher muss dieser Wert nun entsprechend bitweise verschoben werden, damit wir ihn auslesen können => 0000010. Dieser Wert entspricht dem dezimalen Wert 2. Wir lesen nun die Farbe mit dem Index 2 aus der Farbtabelle aus (die Farbtabelle ist im ASCII Teil beschrieben) und erhalten hier die Farbe 255/255/255 nach dem RGB Verfahren. Diese Farbe entspricht der Farbe Weiß. Die restlichen Bit dieses Bytes gehören zur Lauflänge (hier die letzten 3 Bit).

- Kommen wir nun zu den restlichen Werten 90 89 2C 0B aus unserer Bildzeile.

  90 => 10010000 (die roten Bits gehören zur Farbkodierung, die grünen Bits zur Lauflänge)

  89 => 10001001

  2C => 00101100

  0B => 00001101

  Aufgrund der Weiterverkettung ergeben sich hier folgend Lauflängen mit ihren spezifischen Farben

- erstes Pixel mit Lauflänge 0001000100100101100 mit Farbindex (0010)

- zweites Pixel mit Lauflänge 101 mit Farbindex (0001)

ausgewertet befinden sich nun folgende Bildpixel in dieser Bildzeile:

- 1196 Pixel mit der Farbe (255/255/255 RGB) weiß

- 5 Pixel mit der Farbe (0/0/0) schwarz

Zur Überprüfung kann man die Anzahl der Lauflängen addieren und erhält den Wert 1201, der mit der Auflösung aus dem ASCII Teil verglichen (1201) somit einen korrekten Wert liefert. Diese Daten werden dann in einer For Schleife zu unkomprimierten Farb-und Pixelinformationen zusammen gesetzt und entsprechend der Java Konventionen dargestellt.

Anmerkung:

Es können aber auch Nullbytes innerhalb einer Bildzeile auftauchen. Da es sich

hier um sogenannten Datenmüll handelt, werden die Nullbytes zwar als Pixel mitgezählt, bekommen jedoch keine Farbzuordnung. Dieses Problem ist auch in dem Readme des NOAA Chartprojektors beschrieben, der als Hilfsmittel für dieses Projekt zur Verfügung gestellt wurde. Daraus ergibt sich die Tatsache das auch der Autor des NOAA Chartprojektors die ähnliche Probleme beim Einlesen der KAP-Dateien im ASCII und Binärteil hatte. Woher diese Inkonsistenz der Daten im Bezug zu den Dateiformatbeschreibungen stammt, ist leider nicht in Erfahrung zu bringen.

ACHTUNG:

Sollten die Auflösungsvariablen pixelY und pixelX nicht mit der tatsächlichen Anzahl der Daten übereinstimmen, so ist in im Programm keine Fehlerroutine vorgesehen. D.h. die in dem ASCII Datenteil beschriebenen Werte hinsichtlich der Auflösung sind für das Programm bei der Dekomprimierung bindend. Laut Dateiformatbeschreibung sind auch keine anders lautenden Fälle beschrieben. Des weiteren wurde auf die Verwendung der Strip Offset Tabelle verzichtet, da in dieser Anwendung dadurch kein Vorteil entsteht.

Zusammenfassung:

Im Bereich der Dekomprimierung werden folgende Verarbeitungsschritte ausgeführt:

- Dekodieren der Zeilennummer
- Dekodieren der eigentlichen Bildpixel, die durch RLE (RunLengthEncoding) kodiert sind.
- Übertragen der dekodierten Daten in das Zielraster

5.1.3. Package Measurement

Das Package Measurement enthält Klassen die zur Durchführung von Messungen und Berechnungen innerhalb der Seekarten.

Hier nun die einzelnen Klassen mit Ihren Aufgabenstellungen:

Klasse	***Aufgabenstellung***
Coordinate	Datenklasse einer Koordinate
Course	Berechnung eines Kurswinkels
Distance	Berechnung einer Distanz zwischen Abfahrtsort und Bestimmungsort
NavCal	Berechnungsverfahren der nautischen Navigation
ValueWindow	Darstellung der Ergebnisse der jeweiligen Berechnungen

5.1.3.1. Klasse Coordinate

Die Klasse Coordinate stellt eine der Basisklassen dieses Projektes dar. Sie stellt eine einfache Koordinate dar, die aber von vielen anderen Klassen beerbt wird.

Als Parameter besitzt die Klasse Coordinate:

- einen X Wert, der auch Longitude (Längengrad) genannt wird
- einen Y Wert, der auch Latitude (Breitengrad) genannt wird.

5.1.3.2. Klasse Course

Die Klasse Course enthält die Berechnung von Kurswinkeln. Die Darstellung des Kurspfeiles in der jeweiligen Seekarte ist in der Klasse ImagePane implementiert, die einzelnen Berechnungsverfahren sind in der Klasse NavCal vorgesehen. In der Klasse Course werden die verschiedenen Berechnungsverfahren verwaltet, da anhand der vorliegenden Ausgangsdaten (die der Benutzer mit der Maus auf der jeweiligen Seekarte angibt) jeweils verschiedene Berechnungsverfahren gewählt werden müssen (siehe Kapitel Nautische Berechnungsverfahren und Terrestrische Besteckrechnung: Mittelbreitenverfahren / Grosskreisnavigation). Bei kleinen Entfernungen wird das Mittelbreitenverfahren (Breitengradunterschied < 5° und Entfernungen < 600 sm), bei größeren das Verfahren der Grosskreisnavigation gewählt. Die Wahl der Berechnungsmethode hängt somit auch vom Maßstab der vorliegenden Seekarte ab. Die Software GPSMan besitzt somit eine automatische Auswahl der Berechnungsverfahren, kann aber auch seitens des Benutzers über *Optionen / Einstellungen* manuell auf ein bestimmtes Berechnungsverfahren eingestellt werden.

Hier nun das Auswählverfahren:

```
private void calKurs ()
  {
        // automatische Wahl der Berechnungsmethode
        if (prefData.isAutoCalTyp ())
        {
              // Wenn die Breitengraddifferenz kleiner als 5°, dann
Mittelbreitenberechnung
              if (Math.abs (point1.getLatitudeDeg () -
                                    point2.getLatitudeDeg ()) < 5)
              {
                    kurs = NavCal.calMittelbreite (point1, point2,"course");
                    prefData.setMeasCalType(1);
                    // wenn aber die Distanz grösser wie 600 sm ist dann
Grosskreisnavigation
                    if (NavCal.calGrossKreis (point1, point2,"course") > 600)
                    {
                          prefData.setMeasCalType(2);
                    }
              }
              // ansonsten Grosskreisnavigation
              else
              {
                    kurs = NavCal.calGrossKreis (point1, point2,"course");
                    prefData.setMeasCalType(2);
```

```
                        }
                }
                else
                {
                        // Berechnungsverfahren Mittelbreite
                        if (prefData.getMeasCalType () == 1)
                        {
                                kurs = NavCal.calMittelbreite (point1, point2,"course");
                        }
                        // Berechnungsverfahren Grosskreisnavigation
                        else
                        {
                                kurs = NavCal.calGrossKreis (point1, point2,"course");
                        }
                }
        }
```

Die nun ausgewählte Berechnung wird anhand von Methoden in der Klasse NavCal ausgeführt und die Ergebnisse werden in Zeichenkette umgewandelt und stehen mithilfe der Methode getKursValueString für das Anzeigen im Ergebnisfenster bereit.

```
public String getKursValueString ()
{
   return kursString;
}
```

5.1.3.3. Klasse Distance

Ähnlich wie die Klasse Course so ist auch die Klasse Distance lediglich zur Verwaltung der Berechnungsverfahren vorgesehen. Auch hier hängen die genutzten Berechnungsverfahren von der Güte der Ausgangsdaten (siehe Kapitel Klasse Course) ab. Wobei die Klasse Distance unterschiedlich zur Klasse Course noch zusätzlich in der Anzeige (und somit auch in der Berechnung) zwischen der Berechnung von Seemeilen [sm] und Kilometer [km] unterscheiden kann, hierzu ist aber eine entsprechender Eintrag des Benutzers unter *Optionen / Einstellungen* erforderlich.

Hier nun das Auswählverfahren der Klasse Distance:

```
/**
 * calDistance  Berechnet die Entfernung zwischen zwei Punkten
 */
private void calDistance ()
{
   if (prefData.getMeasType () == 1)  // in Seemeilen
   {
      // automatische Wahl der Berechnungsmethode
      if (prefData.isAutoCalTyp ())
      {
         // Wenn die Breitengraddifferenz kleiner als 5°, dann Mittelbreitenberechnung
         if (Math.abs (point1.getLatitudeDeg () -
                  point2.getLatitudeDeg ()) < 5)
         {
            distance = NavCal.calMittelbreite (point1, point2,"dist");
            prefData.setMeasCalType (1);

            // wenn aber die Distanz grösser wie 600 sm ist dann Grosskreisnavigation
            if (distance > 600)
            {
```

```
                distance = NavCal.calGrossKreis (point1, point2,"dist");
                prefData.setMeasCalType (3);
            }
        }

        // ansonsten Grosskreisnavigation
        else
        {
            distance = NavCal.calGrossKreis (point1, point2,"dist");
            prefData.setMeasCalType (3);
        }
    }
    else
    {
        // Berechnungsverfahren Mittelbreite
        if (prefData.getMeasCalType () == 1)
        {
            distance = NavCal.calMittelbreite (point1, point2,"dist");
        }

        // Berechnungsverfahren Grosskreisnavigation
        else
        {
            distance = NavCal.calGrossKreis (point1, point2,"dist");
        }
    }
}
else
// automatische Wahl der Berechnungsmethode
if (prefData.isAutoCalTyp ())
{
    // Wenn die Breitengraddifferenz kleiner als 5Â°, dann Mittelbreitenberechnung
    if (Math.abs (point1.getLatitudeDeg () - point2.getLatitudeDeg ()) < 5)
    {
        distance = NavCal.calSMtoKM (NavCal.calMittelbreite (point1, point2,"dist"));
        prefData.setMeasCalType (1);

        // wenn aber die Distanz grÃ¶sser wie 600 sm ist dann VergrÃ¶sserte Breite
        if (distance > 600)
        {
            distance =NavCal.calGrossKreis (point1, point2,"dist");
            prefData.setMeasCalType (2);
        }
    }

    // ansonsten Grosskreisnavigation
    else
    {
        distance = NavCal.calSMtoKM (NavCal.calGrossKreis (point1, point2,"dist"));
        prefData.setMeasCalType (2);
    }
}
else
{
    // Berechnungsverfahren Mittelbreite
    if (prefData.getMeasCalType () == 1)
    {
        distance = NavCal.calSMtoKM (NavCal.calMittelbreite (point1, point2,"dist"));
    }

    // Berechnungsverfahren Grosskreisnavigation
    else
```

```
        {
           distance = NavCal.calSMtoKM (NavCal.calGrossKreis (point1, point2,"dist"));
        }
      }
    }
```

Wiederrum ähnlich wie in der Klasse Course so werden auch hier die Ergebnisse in Zeichenkette umgewandelt und und stehen mithilfe der Methode getDistanceValueString für das Anzeigen im Ergebnisfenster bereit.

```
/**
 * getDistanceValueString liefert das Ergebnis der Berechnung als String
 *
 * @return
 */
public String getDistanceValueString ()
{
            updateDistance();
   return distString;
}
```

5.1.3.4. Klasse NavCal

Die Klasse NavCal enthält alle wichtigen nautischen Berechnungsverfahren, die zur Messung oder/und Berechnung innerhalb der Software GPSMan benötigt werden.

- **Mittelbreitenverfahren**

Entsprechend der Formel (siehe Kapitel Nautische Berechnungsverfahren und Terrestrische Besteckrechnung) :

Berechnung des Kurs über Grund (KüG):

$ά = \text{arc tan} \{ (\lambda_A - \lambda_B) * \cos [(\varphi_A + \varphi_B) : 2] : (\varphi_A - \varphi_B) \}$

Berechnung der Distanz über Grund (DüG):

$DüG = (\varphi_A - \varphi_B) * 60 : \cos (KüG)$

$DüG = (\lambda_A - \lambda_B) * \cos [(\varphi_A + \varphi_B) : 2] * 60 : \sin (KüG)$

```
/**
 * calMittelbreite  Berechnet die Entfernung  zwischen zwei Punkten in Seemeilen nach dem
 * Verfahren der Mittelbreite
 *
 * @param a Koordinate A
 * @param b Koordinate B
 * @param returnType "dist" für Distanzen, "course" für Kurs und Winkelangaben
 *
 * @return Nach returnType gerichtet, aber double
 */
public static double calMittelbreite (Coordinate a, Coordinate b, String returnType)
{
   double Db;  // Differenz der Breitengrade
   double Dl;  // Differenz der Längengrade
   double Ab;  // Addition der Breitengrade
   double alpha;  //Kurs oder Winkel
```

```
    double distanz;  //Distanz zwischen Punkt A und Punkt B

    // zur Vereinfachung  hier ein paar Vorberechnungen
    Dl = a.getLongitude () - b.getLongitude ();
    Db = a.getLatitude () - b.getLatitude ();
    Ab = a.getLatitude () + b.getLatitude ();

    // Berechnung des Kurswinkel, rechtsweisend von Nord
    alpha = bogWink (Math.atan (Dl * (Math.cos (winkBog (Ab / 2))) / Db));


//                                1.Quadrant
    if (getGeoQuadrant (a, b) == 1)
    {
      // alpha bleibt alpha
    }

//                                2.Quadrant
    else if (getGeoQuadrant (a, b) == 2)
    {
      alpha = alpha + 180;
    }

//                                3.Quadrant
    else if (getGeoQuadrant (a, b) == 3)
    {
      alpha = alpha + 180;
    }

//                                4.Quadrant
    else
    {
      alpha = alpha + 360;
    }


    // Berechnen der Distanz
    // In der Nähe von 90° und 270° wird eine andere Formel verwendet, da Differenz
    // der Breitengrade zu gering wird
    if (((alpha > 90) && (alpha < 100)) || (alpha > 260) && (alpha < 280))
    {
      distanz = Dl * Math.cos (winkBog (Ab) / 2) * 60 / Math.sin (winkBog (alpha));
    }
    else
    {
      distanz = (Db * 60) / Math.cos (winkBog (alpha));
    }

    if (returnType == "dist")
    {
      return Math.abs (distanz);
    }
    else  // course
    {
      return alpha;
    }
  }
```

- **Grosskreisnavigation**

Entsprechend der Formel (siehe Kapitel Nautische Berechnungsverfahren und Terrestrische Besteckrechnung) :

$$d_{GK} = \{\arccos [\sin(\varphi_A) * \sin(\varphi_B) + \cos(\varphi_A) * \cos(\varphi_B) * \cos(\lambda_A - \lambda_B)] \} * 60$$

$$AK = \arccos \{ [\sin(\varphi_B) - \sin(\varphi_A) * \cos(d_{GK} : 60)] : [\cos(\varphi_A) * \sin(d_{GK} : 60)] \}$$

```
/**
 * calGrossKreis  Berechnet die Entfernung  zwischen zwei Punkten in Seemeilen nach dem
 * Verfahren der GrossKreisnavigation
 *
 * @param a Koordinate A
 * @param b Koordinate B
 * @param returnType "dist" für Distanzen, "curse" für Kurs und Winkelangaben
 *
 * @return Nach returnType gerichtet, aber double
 */
public static double calGrossKreis (Coordinate a, Coordinate b, String returnType)
{
    double Ay = a.getLatitude ();
    double Ax = a.getLongitude ();
    double By = b.getLatitude ();
    double Bx = b.getLongitude ();

    //                    Differenz der Längengrade
    double Dl = Ax - Bx;

    // Berechnen der Distanz
    double distanz = bogWink (Math.acos (Math.sin (winkBog (Ay)) * Math.sin (winkBog (By)) +
                           Math.cos (winkBog (Ay)) * Math.cos (winkBog (By)) * Math.cos (winkBog
(Dl)))) * 60;

    // Berechnen des Kurswinkel
    double alpha = bogWink (Math.acos ((Math.sin (winkBog (By)) -
                          Math.sin (winkBog (Ay)) * Math.cos (winkBog (distanz / 60))) / (Math.cos
(winkBog (Ay)) * Math.sin (winkBog (distanz / 60)))));

                        // Unterscheidung in östlicher und westlicher Halbkreis
                        if (Ax < Bx)
                        {
                                    //alpha bleibt alpha
                        }
                        else
                        {
                                    alpha = 360 - alpha;
                        }

    if (returnType == "dist")
    {
        return distanz;
    }
    else  //course
    {
        return alpha;
    }
}
```

- **Umrechnung von Seemeilen in Kilometer**

Zur Darstellung der Ergebnisse müssen die Seemeilen noch in Kilometer umgerechnet werden:

```
/**
 * calSMtoKM Rechnet von Seemeilen auf Kilometer um.
```

```
 *
 * @param a Seemeile
 *
 * @return Kilometer
 */
public static double calSMtoKM (double a)
{
   return a * 1.852
}
```

- **Auswahl der entsprechenden Quadranten**

Aufgrund der Berechnungsverfahren müssen zur Berechnung der Winkel auch noch die korrekten Quadranten bestimmt werden. Hierzu wurde die Methode getGeoQuadrant implementiert:

```
   /**
    * getQuadrant die Methode getGeoQuadrant ermittelt den für einen Winkel
    * den erforderlichen Quadranten, Breiten- und LÃ¤ngengradsystem
    *
    * @param a Koordinate A
    * @param b Koordinate B
    *
    * @return Nummer des Quadranten als int Variable
    */
   public static int getGeoQuadrant (Coordinate a, Coordinate b)
   {
//                              1.Quadrant
      if ((a.getX () < b.getX ()) && (a.getY () <= b.getY ()))
      {
         return 1;
      }

//                              2.Quadrant
      else if ((a.getX () <= b.getX ()) && (a.getY () > b.getY ()))
      {
         return 2;
      }

//                              3.Quadrant
      else if ((a.getX () > b.getX ()) && (a.getY () >= b.getY ()))
      {
         return 3;
      }

//                              4.Quadrant
      else
      {
         return 4;
      }
   }
}
```

Eine weitere Methode zur Auswahl der Quadranten ist in der Klasse NavCal enthalten, diese basiert aber auf einem anderen Koordinatensystem und dient der Darstellung des Kurspfeiles in der Klasse ImagePane. Der Struktur wegen gehört Sie aber in die Klasse NavCal. Daher erfolgt ihre Beschreibung in der Klasse ImagePane.

- **Bestimmung der Himmelsrichtung**

Entsprechend des Winkelwertes kann in der Methode getDirectionString die Himmelsrichtung zur Darstellung im Ergebnisfenster bestimmt werden:

```
/**
 * getDirectionString diese Methode liefert fÃ¼r einen entsprechenden  Winkel die
 * entsprechenden nautischen Bezeichnungen (NW, SE, usw.)
 *
 * @param alpha Winkel in Grad
 *
 * @return Bezeichnung des Winkels als String
 */
public static String getDirectionString (double alpha)
{
        if ((alpha >=0)&&(alpha<22.5))
        {
                return "N";
        }
        else if ((alpha >=22.5)&&(alpha<67.5))
        {
                return "NE";
        }
        else if ((alpha >=67.5)&&(alpha<112.5))
        {
                return "E";
        }
        else if ((alpha >=112.5)&&(alpha<157.5))
        {
                return "SE";
        }
        else if ((alpha >=157.5)&&(alpha<202.5))
        {
                return "S";
        }
        else if ((alpha >=202.5)&&(alpha<247.5))
        {
                return "SW";
        }
        else if ((alpha >=247.5)&&(alpha<292.5))
        {
                return "W";
        }
        else if ((alpha >=292.5)&&(alpha<337.5))
        {
                return "NW";
        }
        else
        {
                return "N";
        }
}
}
```

5.1.3.5. Klasse ValueWindow

Die Klasse ValueWindow stellt ein Ausgabemedium in Form eines Dialogfensters für die Messungen und Berechnungen innerhalb der Software GPSMan dar. Bei jeder neuen Berechnung wird dieses Fenster akualisiert und wieder dargestellt. Der Benutzer ist jedoch in der Lage das Fenster manuell über einen entsprechenden Menüeintrag und auch Button zu schliessen.

5.1.4. Package Tracking

Package	***Aufgabenstellung***
GPS	Leeres Package zur späteren Anbindung von GPS Geräten
Sim	Simulation von Bewegungsdaten
Symbol	Symbolklassen für Trackpunkt Symbole

Klassen	***Aufgabenstellung***
ShipInfo	Darstellen der aktuellen Schiffsposition in der Seekarte
TrackDialog	Dialog zum Bearbeiten der Tracks
TrackingAdapter	Interface für alle Adapterklassen
TrackingDisplayDialog	Darstellen der aktuellen Bewegungsdaten des Schiffes
Trackpoint	Datenklasse der Trackpunkte
TrackpointDetailPanel	Panel mit den elementaren Daten der Trackpunkte
TrackpointManageDialog	Dialog zum Verwalten von Trackpunkten und Tracks
TrackpointManager	Diese Klasse enthält Methoden zum Bearbeiten der Trackpunkten
TrackpointTableModel	Modellklasse zum Darstellen der Trackpunkte in Listen

5.1.4.1. Package GPS

Klasse	***Aufgabenstellung***
GPSTrackingAdapter	Leere Klasse

5.1.4.1.1. Klasse GPSTrackingAdapter

Diese Klasse ist leer und enthält keine Funktionalität. Sie ist für die spätere Anbindung eines GPS Gerätes gedacht und wurde von der Klasse TrackingAdapter abgeleitet.

5.1.4.2. Package Sim

Klasse	***Aufgabenstellung***
SimData	Datenklasse der Startparameter der Simulation
SimStartDialog	Dialog zum Eingeben der Startparameter
SimSteerPanel	Steuerungsdialog der laufenden Simulation
SimTrackingAdapter	Berechnung der simulierten Schiffsbewegungen

5.1.4.2.1. Klasse SimData

Die Klasse SimData ist die Datenklasse der Startparameter, die für die Berechnung der Simulation erforderlich sind.

5.1.4.2.2. Klasse SimStartDialog

Die Klasse SimStartDialog enthält ein grafisches Dialogfenster, in dem die Startparameter der Simulation eingegeben werden müssen. Diese Startparameter sind wie folgt:

- Geschwindigkeit des Schiffes in Knoten [kn]
- Richtung des Kurses in Grad
- Abweichung durch Gegenwind oder Gegenströmung in Knoten
- Richtung der Abweichung in Grad
- Zeitintervall der Berechnung der Simulationsdaten in Sekunden [sec]

5.1.4.2.3. Klasse SimSteerPanel

Die Klasse SimSteerPanel enthält die grafische Benutzeroberfläche zur Steuerung des simulierten Schiffes. In ihr können während der laufenden Simulation die Parameter geändert werden. Diese geänderten Werte werden der Klasse SimTrackingAdapter zugeführt und die Berechnung der Simulationsdaten wird dort entsprechend angepasst.

5.1.4.2.4. Klasse SimTrackingAdapter

In der Klasse SimTrackingAdapter werden die Schiffsbewegungen simuliert und die dadurch entstehenden Werte werden dem Programm in der gleichen Form wie „echte“ GPS Daten zur Darstellung zur Verfügung gestellt.

Hier das Kernelement der Simulation, die Berechnung der Schiffsbewegungen die in festen Zeitintervallen durchgeführt wird. Das Zeitintervall wird als Parameter beim Start der Simulation seitens des Benutzer festgelegt:

Die Berechnung der tatsächlich verstrichenen Zeit seit der letzten Berechnung, die minimal vom vorgegebenen Zeitintervall abweichen kann.

```
// Intervall in Sekunden
double timeInterval = ((double) (currentTrackpoint.getTimeStamp ().getTime () -
                lastTrackpoint.getTimeStamp ().getTime ())) / 1000.0;
```

Die Berechnung der zurückgelegten Distanz innerhalb eines Zeitintervalls (ohne Abweichung durch Gegenwind und Gegenströmung).

```
// Zurückgelegte Distanz in Seemeilen
double distance = (simData.getSpeed () / 3600.0) * timeInterval;
```

Die Umrechnung des nautischen Winkels (0° ist Nord)in einen mathematischen Winkel (0° ist X-Achse, entspricht West).

```
// Kurs in mathematischen Winkel und Bogenmass umrechnen
double course = (360.0 - (simData.getCourse () - 90.0));
course -= (((long) course) / 360) * 360;
course = Math.toRadians (course);
```

Die Berechnung der zurückgelegte Distanz durch die Abweichung durch Gegenwind oder / Gegenströmung.

```
// Zurückgelegte Distanz in Seemeilen (Drift)
double distanceDrift = (simData.getSpeedDrift () / 3600.0) * timeInterval;
```

Die Umrechnung des nautischen Winkels (0° ist Nord) der Abweichung in einen mathematischen Winkel (0° ist X-Achse, entspricht West).

```
// Kurs in mathematischen Winkel und Bogenmass umrechnen (Drift)
double courseDrift = (360.0 - (simData.getDirectionDrift () - 90.0));
courseDrift -= (((long) courseDrift) / 360) * 360;
courseDrift = Math.toRadians (courseDrift);
```

Vektorielle Addition der beiden Bewegungen.

```
// Resultierende x/y Komponenten
double distancex = Math.cos (course) * distance +
            Math.cos (courseDrift) * distanceDrift;
double distancey = Math.sin (course) * distance +
            Math.sin (courseDrift) * distanceDrift;
```

Berechnung der neuen Position des simulierten Schiffes nach der Mittelbreitenrechnung (siehe Kapitel *Fachliches Umfeld*).

```
// Distanz und neue Position im Gradmass mit Korrektur der Längengrade
distancey /= 60.0;

double newPosLat = lastTrackpoint.getLatitude () + distancey;

// Division durch 0 kann auftreten!
distancex /= 60.0 * Math.cos (Math.toRadians (lastTrackpoint.getLatitude () +
                              newPosLat) / 2.0);

double newPosLong = lastTrackpoint.getLongitude () + distancex;
```

Des weiteren sind in der Klasse SimTrackingAdapter Methoden zum Starten und Stoppen der Simulation enthalten.

5.1.4.3. Package Symbol

Package	***Aufgabenstellung***
SymbolListModelTP	Modellklasse für die Darstellung der Trackpunkte in Listen und Comboboxen

5.1.4.3.1. Klasse SymbolListModelTP

Die Klasse SymbolListModelTP ist das Modell für Listen und Comboboxen der Trackpunktsymbole.

5.1.4.4. Klasse ShipInfo

Die Klasse ShipInfo verfolgt die Änderungen der Schiffsposition.

5.1.4.5. Klasse TrackDialog

Die Klasse TrackDialog ist die grafische Benutzeroberfläche zum Erstellen und Ändern von Tracks.

5.1.4.6. Klasse TrackingAdapter

Die abstrakte Klasse TrackingAdapter definiert ein Interface, an das sich alle Adapterklassen (GPSTrackingAdapter, SimTrackingAdapter) halten müssen. Alle Adapterklassen müssen von dieser Klasse abgeleitet werden.

5.1.4.7. Klasse TrackDisplayDialog

Die Klasse TrackDisplayDialog stellt ein Dialogfenster mit den aktuellen Parametern der Schiffsposition zur Verfügung.

Hier die Parameter:

- Latitude (Breitengrad)
- Longitude (Längengrad)
- Geschwindigkeit in Konten [kn]
- Kurs in nautischen Grad
- Datum
- Bordzeit

Dieses Dialogfenster ist in verschiedenen Grössen zur ständigen Information des Schiffführers gedacht, so das dieser auch aus einiger Entfernung die aktuellen Positionsdaten visuell entnehmen kann.

5.1.4.8. Klasse Trackpoint

Die Klasse Trackpoint stellt die Datenklasse eines einzelnen Trackpunktes dar

und ist von der Klasse Waypoint abgeleitet.

Weitere Merkmale sind:

- Geschwindigkeit
- Kurs
- Zeitpunkt der Datenerfassung
- Farbe

5.1.4.9. Klasse TrackpointDetailPanel

Die Klasse TrackpointDetailPanel enthält die grafische Benutzeroberfläche zum Anzeigen aller Daten eines Trackpunktes. Dieses Panel wird zum Anzeigen in anderen Klassen benutzt.

5.1.4.10. Klasse TrackpointManageDialog

In dieser Klasse werden alle Elemente, die zur Verwaltung der Trackpunkte und Tracks notwendig sind, als grafische Schnittstelle zum Benutzer dargestellt. Als weiteres Element wird ein TrackpointDetailPanel zur Darstellung der elementaren Trackpunktparameter mit eingebunden. Des können hier die Methoden zum Laden und Speichern von Trackpunktdateien aufgerufen werden.

5.1.4.11. Klasse TrackpointManager

Die Klasse TrackpointManager beinhaltet die Methoden zum Laden und Speichern der Trackpunkte und Tracks in XML Dateien. Sie enthält die Liste aller Trackpunkte und die Liste aller Tracks. Die Methoden zum Bearbeiten der jeweiligen Liste (Trackpunkte und Tracks) sind auch Bestandteil dieser Klasse.

5.1.4.12. Klasse TrackpointTableModel

Da die Trackpunkte in der Klasse TrackpointManageDialog in einer Tabelle gezeigt werden, benötigt diese Tabelle ein TableModel zur Darstellung der Trackpunktdaten.

5.1.5. Package Util

Klasse	***Aufgabenstellung***
BulletinLayout	Layoutmanager
DigitTextField	Eingabefeld für Ziffern
NumberTextField	Eingabefeld für Zahlen
Tool	Universelle statische Hilfsklasse
WaypointTextField	Eingabefeld für Wegpunktnamen

5.1.5.1. Klasse BulletinLayout

Die Klasse BulletinLayout ist ein weiterer Layoutmanager, der für die Darstellung des Tracking Dialoges in der Klasse TrackingDialog verwendet wurde. Bei der Nutzung von dem BulletinLayout können absolute Werte zum Positionieren von grafischen Elementen verwendet werden.

5.1.5.2. Klasse DigitTextField

Die Klasse DigitTextField ist ein Eingabefeld für Ziffern. Die Tastatureingabe des Benutzer wird abgefangen, untersucht und entsprechend den Restriktionen an das Eingabefeld weitergeleitet. Bei der Klasse DigitTextField sind bei der Eingabe nur Ziffern und die Zeichen „+“ und „-“ erlaubt.

```
public void insertString (int offs, String str, AttributeSet a)
            throws BadLocationException
  {
    // die Eingabe wird abgefangen und untersucht, nur Ergebnisse
    // die zugelassen sind werden an das Eingabefeld weitergeleitet
    char source[] = str.toCharArray ();
    char result[] = new char[source.length];
    int j = 0;

    for (int i = 0; i < result.length; i++)
    {
      if (Character.isDigit (source[i]))
      {
        result[j++] = source[i];
      }
      else if (source[i] == '+')
      {
        if (firstPosition ())
        {
          result[j++] = source[i];
        }
      }
      else if (source[i] == '-')
      {
        if (firstPosition ())
        {
          result[j++] = source[i];
        }
      }
      else
      {
        toolkit.beep ();
      }
    }

    super.insertString (offs, new String(result, 0, j), a);
  }
}
```

5.1.5.3. Klasse NumberTextField

Ähnlich wie bei der vorherigen Klasse DigitTextField, so handelt es sich bei der Klasse NumberTextField auch um ein Eingabefeld, dessen Tastatureingaben abgefangen und bewertet werden. Zusätzlich zur Funktionalität bei der Klasse DigitTextField ist hier die Eingabe eines mathematischen Komma und eines

Punktes (als Komma) erlaubt. Bei der Eingabe eines mathematischen Komma wird dieses automatisch in ein Zeichen eines Punkte umgewandelt.

Mit Verfahren dieser Art werden Fehlereingaben seitens der Benutzer sehr stark eingeschränkt.

5.1.5.4. Klasse Tool

In der Klasse Tool sind einzelne Methoden abgelegt, die mehrfach im Projekt benötigt werden. Damit ist die Tool Klasse als eine universelle Hilfsklasse zu bezeichnen, die damit zwar dem objektorientieren Ansatz nicht entspricht, aber redundanten Sourcecode eliminiert.

5.1.5.5. Klasse WaypointTextField

Die Klasse WaypointTextField beschränkt (ähnlich wie in NumberTextField) die Eingabe von Namen von Wegpunkten. Folgende Restriktionen gelten in dieser Klasse:

- es sind alle Buchstaben zugelassen, wobei alle Kleinbuchstaben in Grossbuchstaben automatisch umgewandelt werden.
- es sind alle Ziffern zugelassen
- es sind Leerzeichen zugelassen

5.1.6. Package Waypoint

Package	***Aufgabenstellung***
Symbol	Symbolklassen für Wegpunkt Symbole

Klasse	***Aufgabenstellung***
Route	Datenklasse der Routen
RouteComboModel	Modellklasse zur Darstellung der Routen in Comboboxen
RouteDialog	Dialog zum Erstellen von Routen
RouteListModel	Modellklasse zur Darstellung der Routen in Listen
Waypoint	Datenklasse der Wegpunkte
WaypointDetailPanel	Panel zur Darstellung der elementaren Daten der Wegpunkte
WaypointDialog	Dialog zum Bearbeiten der Wegpunkte
WaypointManageDialog	Dialog zum Verwalten von Wegpunkten und Routen
WaypointManager	Diese Klasse enthält Methoden zum Bearbeiten der Wegpunkten
WaypointTableModel	Modellklasse zur Darstellung von Wegpunkten in Listen

5.1.6.1. Package Symbol

Package	***Aufgabenstellung***
SymbolListCellRenderer	Zeichnet ein Symbol in Listen und Comboboxen
SymbolListModel	Interface für alle ListModelklassen
SymbolListModelWP	Modellklasse für die Darstellung der Wegpunkte in Listen und Comboboxen
SymbolTableCellRenderer	Zeichnet ein Symbol in eine Tabellenzelle

5.1.6.1.1. Klasse SymbolListCellRenderer

Die Klasse SymbolListCellRenderer zeichnet ein Symbol (Trackpunkt oder Wegpunkt) in eine Listen- oder Comboboxzelle.

5.1.6.1.2. Klasse SymbolListModel

Die Klasse SymbolListModel ist eine abstrakte Basisklasse für alle SymbolListen-Modelle.

5.1.6.1.3. Klasse SymbolListModelWP

Die Klasse SymbolListModelWP ist das Modell für Listen und Comboboxen der Wegpunktsymbole.

5.1.6.1.4. Klasse SymbolTableCellRenderer

Die Klasse SymbolListCellRenderer zeichnet ein Symbol (Trackpunkt oder Wegpunkt) in eine Tabellenzelle.

5.1.6.2. Klasse Route

Die Klasse Route enthält alle Daten zu einer Route oder einem Track.

Hier die Elemente einer Route oder eines Tracks:

- Name
- Notiz
- Farbe
- Liste von Wegpunkten (Trackpunkte sind auch Wegpunkte)

Die Methoden zur Zuordnung der Wegpunkte rufen nur ihre Äquivalente in der Klasse Waypoint auf.

5.1.6.3. Klasse RouteComboModel

Dies ist die Modellklasse die die Darstellung von Routen in Comboboxen

ermöglicht.

5.1.6.4. Klasse RouteDialog

Die Klasse RouteDialog ist die grafische Benutzeroberfläche zum Erstellen und Ändern von Routen.

5.1.6.5. Klasse RouteListModel

Dies ist die Modellklasse die die Darstellung von Routen in Listen ermöglicht.

5.1.6.6. Klasse Waypoint

Die Klasse Waypoint stellt die Datenklasse eines einzelnen Wegpunktes dar und ist von der Klasse Coordinate abgeleitet. Durch die Vererbung sind die Längen- und Breitengradangaben schon aus der Klasse Coordinate vorgegeben. Weitere Merkmale sind:

- der Name des Wegpunktes
- die Notiz zum Wegpunkt
- das Symbol des Wegpunktes
- zugeordnete Routen

In der Methode addRoute() wird ein Verweis zur einer Route hinzugefügt. Gleichzeitig wird in der Route eine Referenz auf diesen Wegpunkt gespeichert.

```
public void addRoute (Route route)
{
   if (route != null)
   {
      routeSet.add (route);
      route.assignWaypoint (this);

      if (routeSet.size () == 1)
      {
         WaypointManager.getInstance ().handleFirstWpRouteAssigned (this);
      }
   }
}
```

Hier die Rückwärtsreferenz aus der Klasse Route

```
protected void assignWaypoint (Waypoint wp)
{
   waypoints.add (wp);
}
```

Das Löschen von Zuordnungen funktioniert entsprechend.

5.1.6.7. Klasse WaypointDetailPanel

Die Klasse WaypointDetailPanel enthält die grafische Benutzeroberfläche zum Anzeigen aller Daten eines Wegpunktes. Dieses Panel wird zum Anzeigen in anderen Klassen benutzt.

5.1.6.8. Klasse WaypointDialog

In der Klasse WaypointDialog wird die grafische Schnittstelle zur Erstellung eines Wegpunktes dargestellt. Hier werden die Daten eines einzigen Wegpunktes gezeigt und dem Benutzer wird die Möglichkeit gegeben die Parameter eines Wegpunktes zu bearbeiten und zu verändern.

5.1.6.9. Klasse WaypointManageDialog

In dieser Klasse werden alle Elemente, die zur Verwaltung der Wegpunkte und Routen notwendig sind, als grafische Schnittstelle zum Benutzer dargestellt. Als weiteres Element wird ein WaypointDetailPanel zur Darstellung der elementaren Wegpunktparameter mit eingebunden. Des weiteren können hier die Methoden zum Laden und Speichern von Wegpunktdateien aufgerufen werden.

Als Beispiel hier die Methode zum Laden von Wegpunktdateien:

```
/**
 * loadPointFile  Aufruf des Dateiauswahlmodus (JFileChooser) und laden einer Wegepunkt
 * und Routendatei
 */
protected void loadWaypointFile ()
{
   if (fc == null)
   {
      fc = new JFileChooser(System.getProperty (currentPath));
      fc.setFileFilter (new WpFileFilter());
   }

   if (JFileChooser.APPROVE_OPTION == fc.showOpenDialog (this))
   {
      int oldSize = routeListModel.getSize ();
      wpManager.loadFile (fc.getSelectedFile ().getAbsolutePath ());
      routeListModel.fireStupidChangedEvent (oldSize);
      wpTableModel.fireTableDataChanged ();
      pack ();
   }
}
```

5.1.6.10. Klasse WaypointManager

Die Klasse WaypointManager beinhaltet die Methoden zum Laden und Speichern der Wegpunkte und Routen in XML Dateien. Sie enthält die Liste aller Wegpunkte und die Liste aller Routen. Die Methoden zum Bearbeiten der jeweiligen Liste (Wegpunkte und Routen) sind auch Bestandteil dieser Klasse.

5.1.6.11. Klasse WaypointTableModel

Da die Wegpunkt in der Klasse WaypointManageDialog in einer Tabelle gezeigt werden, benötigt diese Tabelle ein TableModel zur Darstellung der Wegpunktdaten.

5.1.7. Klasse Application

Die Klasse Application ist die Startklasse des Projektes GPS Motion and Navigation. Sie beherbergt die Main Methode zum Starten des Anwendung.

5.1.8. Klasse GpsManFrame

In der Klasse GPSManFrame wird das eigentliche Hauptfenster der Anwendung implementiert. Hierzu gehört der Bereich im dem die eigentliche Seekarte dargestellt wird (wird durch die Klasse ImagePane realisiert) und die dazugehörenden Menüpunkte und Buttonleisten. Des weiteren wird eine Statuszeile angezeigt, die den aktuellen Punkt des Mauszeigers auf der Karte zeigt und sofort die Umrechnung bzw. die kalbrierten Punkte zeigt. Die Klasse GPSManFrame behandelt daher alle Ereignisse die durch Betätigung einer der Menuepunkte oder Buttons ausgelöst werden.

5.1.9. Klasse ImagePane

Die Klasse ImagePane beinhaltet Methoden zum:

- Laden von Images (Seekarten)
- Verkleinern und Vergrössern der Bilddateien
- Zeichnen von Objekten das jeweilige Image
    - Einzelnen Wegpunkten (in verschiedenen Symbolen)
    - Wegpunktlisten und Routen (in verschiedenen Symbolen)
    - Trackpunkten und Tracks (in verschiedenen Symbolen)
    - der aktuellen Schiffsposition (Fähnchen mit Kurspfeil)
    - Messlinien (Distanzmessungen)
    - Kursberechnung (Kurspfeil)

In dieser Klasse finden alle Zeichenoperationen in der Seekarte statt.

5.1.10. Klasse PrefData

Die Klasse PrefData ist die Datenklasse der Optionen oder Einstellungsdaten bezüglich unseres Projektes GPS Motion and Navigation. Hier eine kurze Auflistung der Einstellungsdaten:

- Der Zoomfaktor
- Einstellungen zu den Wegpunkte
- Einstellungen zu den Trackpunkte

- Einstellungen GPS
- Messungen
- allgemeine Symbole

5.1.11. Klasse PrefDialog

Die Klasse PrefDialog ist die Viewklasse der Optionen oder Einstellungen. Sie stellt die verschiedenen Möglichkeiten der einzelnen Einstellungen in verschiedenen Registerkarten zur Verfügung.

6. Systemtest

6.1. Testmethoden

Anhand von Grundstrategien des White-Box Testverfahren wurden alle Möglichkeiten der Software meinerseits abgeprüft.

6.2. Testergebnisse

6.2.1. Systemfehler

Während den Testphase und auch während der Implementierung sind einige Fehler aufgetreten, die von meiner Seite aus nicht behoben und/oder beeinflusst werden können.

- Beim Benutzen der JDK Version 1.4.0 kann es vorkommen, das eingescannte Seekarten nicht korrekt vom Programm dargestellt werden. Besonders im rechten Bereich wird ein Teil (etwa 25% des Bildes) nur als grafisches Rauschen dargestellt. Nach intensiver Überprüfung, unteranderem mit einer JDK Version 1.4.0 unter Linux, kam ich zu dem Schluss, das hier ein Problem der Windowsversion von Java vorliegen muss. Die Linuxvariante zeigte das Image immer in einwandfreier Qualität.

- Aufgrund des vorherigen beschriebenen Problems, sind auch die Windows® JDK Versionen 1.4.1 und 1.4.2 getestet worden. Bei der Version 1.4.2 sind keine Darstellungsfehler aufgetreten, allerdings ist es offensichtlich das SUN Microsystems® hier an der grafisches Engine gearbeitet hat. Die Darstellungsfehler aus der Version 1.4.0 sind weg, aber der grafische Aufbau beim Bewegen von Seekarten passiert wesentlich langsamer, etwa mit dem Faktor 2-3. Eine entsprechende Anfrage per Email an SUN Microsystems® hinsichtlich dieses Verhaltens ist bis heute unbeantwortet. Somit ist mit einem erheblichen zusätzlichem Zeitaufwand beim Zoomen von Seekarten oder beim Bewegen innerhalb der Seekarten zu rechnen.

- Beim Laden von großen Bilddateien muß die Software GPSMan mit den Argumenten **-Xmx1000000000** gestartet werden. Ansonsten kommt die VM (Virtual Machine) von Java nicht mit den Dateigrößen der Bilddateien zurecht und bricht das Programm mit der Fehlermeldung java.lang.OutOfMemoryError ab. Eine Behebung dieses Umstandes ist mir trotz intensiver Suche nicht gelungen. Eine Recherche im Internet und eine Anfrage per Email bei SUN Microsystems® ergab bisher keine Lösung. Als pragmatische Überbrückung dieses Problemes wurde mit dem oben beschriebenen Argumenten gearbeitet, die Java veranlassen, den maximalen Speicherplatz (heap space) zu reservieren. Das Interssante an diesem Phänomen ist, das trotz der Tatsache das mein Rechner über 512 MB Ram und eine einzelne Bilddatei nicht größer als 15 MB (dies war die größte KAP Seekarte in meinen Beispielkarten), die Fehlermeldung auftritt.

6.2.2. Konzeptfehler

- Beim Kalibrieren von großen Seekarten im KAP Format werden seitens der Seekarten meist viele Referenzpunkte zur Kalibrierung mitgegeben. Wie

schon vorher beschrieben, ist in diesem Projekt nur die Kalibrierung mit dem Drei Punkt Verfahren implementiert. Dadurch werden aber Karten, die verzerrt digitalisiert wurden, hier nicht korrekt kalibriert. Daher muss beachtet werden, das bei großen Seekarten die Kalibrierung ungenau sein kann. Dies aber meist nur in der Mitte der Seekarten, da ein Algorithmus zur Auswahl der drei besten Punkte zur Kalibrierung, immer die am weitesten von einander entfernten Punkte auswählt. Diese drei Punkte liegen also meist in den jeweiligen Ecken der Seekarte, dadurch bedingt die Ungenauigkeit in der Mitte von großen Seekarten (am weitesten von den Kalibrierungspunkten entfernt). Für die Entwicklung einer Kalibriermethode mit mehreren Referenzpunkten reicht der Zeitraum (unter Beachtung der durch das Pflichtenheft bereits beschriebenen Funktionalitäten) dieser Diplomarbeit nicht mehr aus.

7. Bedienungsanleitung

Die GPSMan Anwendung ist nach den allgemeinen Regeln einer Fensteranwendung programmiert worden. Sie besitzt eine Menüleiste und eine Buttonleiste (Toolbar). Die einzelnen Benutzerschnittstellen sind somit über die Buttonleiste und/oder die Menüleiste zu erreichen.

GPSMan wird beim ersten Start nicht im Vollbildmodus gestartet, sondern anhand der Abmessungen der derzeit beim Benutzer eingestellten Auflösung in der Hälfte der Auflösung. Eine Darstellung im Vollbildmodus ist natürlich möglich und kann durch den Benutzer erfolgen.

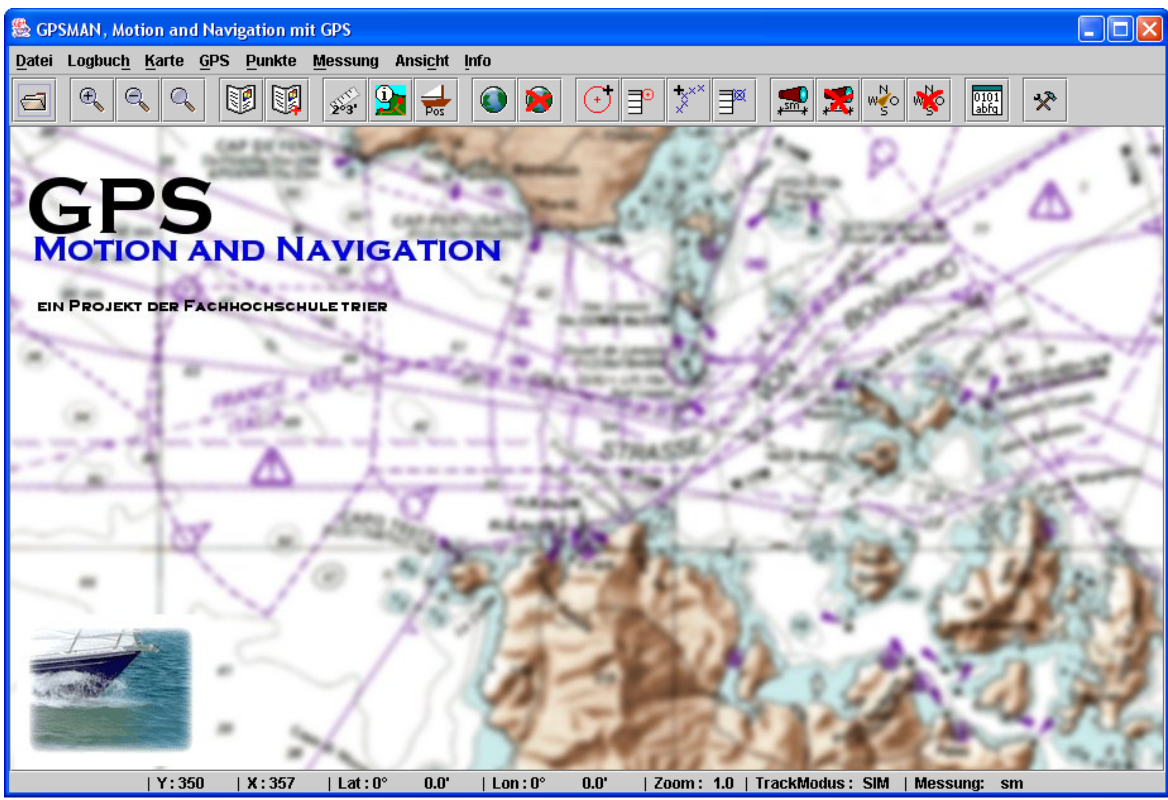


Abbildung 9 das GPSMan Hauptfenster

Hier nun die einzelnen Menüpunkte und ihre Erläuterung in der Anwendung.

7.1. Grundsätzliche Anforderungen

Der Benutzer sollte die Möglichkeit haben Seekarten einzulesen und darzustellen. Mit der Darstellung sollte auch die Möglichkeit vorhanden sein, die Ansicht der Seekarten zu varieren. Dies umfasst die Möglichkeiten die Seekarte in der Ansicht zu verkleinern und zu vergrössern (Zoom), sowie die Ansicht innerhalb der Seekarte nach rechts, links, oben odern unten zu verschieben (scrollen).

- Im GPSMan können Karten aus dem Format *.JPEG und *.KAP einlesen werden.

- Die eingelesenen Karten können nach Belieben gezoomt werden. Es gibt allerdings eine Einschränkung, bei der Verkleinerung der Karten

wird ab dem Skalierungsfaktor von 0,01 % der Originalgröße wird kein weiterer Zoom ausgeführt. Ein Verkleinerung dieser Größenordnung ist nicht sinnvoll und führt zu unnötigem Rechenaufwand.

  - Mithilfe einer Scrollbar kann der Benutzer sich durch Scrollen in alle Himmelsrichtungen auf der jeweiligen Karte bewegen.

  - Der Benutzer muss bzw. kann eingescannte Karten (im JPEG Format) kalibrieren können. Hierzu ist der Menüpunkt *Karte / Kalibrieren* zu benutzen. Diese Kalibrierungsdaten werden automatisch in einer externen Datei gespeichert (<Name der Karte>.cal) werden. Die digitalen Kartendateien des Formats KAP führen ihre Kalibrierungsdaten schon in der Datei mit (siehe Projekt SKKD).

- Der Benutzer sollte die Möglichkeit haben ein Logbuch zu führen (nicht Teil meiner Diplomarbeit). Hier wurde allerdings die Benutzerschnittstelle des Logbuches bereits vorgesehen.

- Der Benutzer soll die Möglichkeit haben offline und online (im Bezug auf ein vorhandenes GPS Gerät) arbeiten zu können. Hierzu kann der Benutzer ein GPS Gerät anschließen (nicht Teil meiner Diplomarbeit) oder den Simulationsmodus der Software starten. In beiden Fällen werden so genannte Tracks und Trackpunkte erzeugt, die auf der Karte den tatsächlich zurück gelegten Weg des Schiffes darzustellen.

- Beim Online Modus kann das GPS Gerät in definierten Zeitabständen, so genannte Trackpunkte auf der Kartenoberfläche erzeugen, die den tatsächlichen Kurs darstellen.

  - Diese Trackpunkte können zu so genannte Tracks zusammen gefasst werden und können in einer externen Datei gespeichert werden.

  - Des weiteren können sie Tracks im einem Verwaltungsmodus nachbearbeitet werden

- Der Benutzer kann so genannte Wegpunkte manuell anlegen.

  - Diese Wegpunkte können zu Routen zusammengefasst werden und können in einer externen Datei gespeichert und später auch wieder geladen werden.

  - Ein Verwaltungsmodus vereinfacht die Bearbeitung der Routen und Wegpunkte

- Der Benutzer soll durch einfaches Anklicken von Positionen in der dargestellten Karte Entfernungen abmessen und Kurse berechnen können.

7.2. Die Menüleiste

Um die eben beschriebenen Benutzerschnittstellen herzustellen, wurde die nun folgende beschriebene Menüstruktur gewählt.

7.2.1. Menü Datei

In dem Menüpunkt finden sich die Möglichkeiten Karten und Bilder zu öffnen, die allgemeinen Einstellungen des Programmes zu bearbeiten und die Anwendung zu beenden.



Abbildung 10 Menü Datei

7.2.1.1. Karte / Bild öffnen

Über den Menüpunkt *Datei / Karte/Bild öffnen* kann der Benutzer vorhandene Dateien (Bildimages im Format JPEG) von Seekarten oder digitale Karten im Format KAP zur Ansicht öffnen. Erwähnen möchte ich jedoch, das GPSMan in der Lage ist jegliche JPG Datei einzulesen, unabhängig davon ob es sich um eine Seekarte handelt oder nicht. Der Menüpunkt *Karte / Bild öffnen* ist zusätzlich über die Tastenkombination STRG-O erreichbar.

Die Dateiauswahl wird über eine standardisiertes Dateiauswahlfenster von Java übernommen. Angezeigt werden in diesem Auswahlfenster standardgemäß nur Dateien mit der Endung *.JPG und *.KAP (so genannte Kartendateien, die Auswahl kann aber vom Benutzer auf alle Dateitypen („Alle Dateien“ unter Dateitypen einstellen) ausgeweitet werden. Allerdings ist dann nicht mehr gewährleistet, wie das Programm auf evtl. andere Dateien reagiert, da nur das Einlesen für die Dateitypen *.JPG und *.KAP korrekt vorgesehen wurde. Unter dem Punkt *„Suchen in“* , so wie in der Mitte des Fenster ersichtlichen Struktur kann eine entsprechende Karte oder Verzeichnis ausgewählt werden.

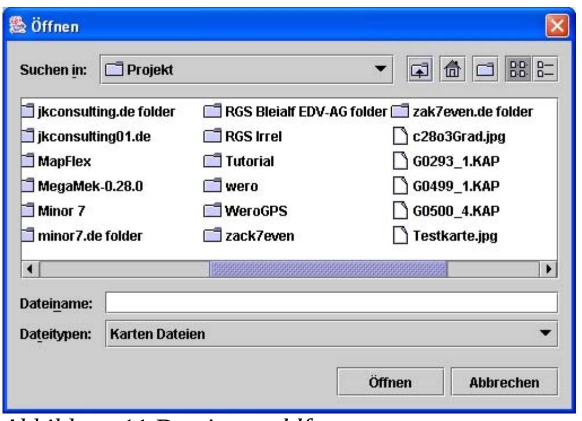


Abbildung 11 Dateiauswahlfenster

- In der oberen rechten Ecke des Dateiauswahlfenster finden sich fünf zusätzliche Buttons, die nun näher beschrieben werden sollen. Der erste Button (von links) führt den Benutzer eine Ebene in der Verzeichnisstruktur nach oben, der zweite Button führt die Anzeige in das Home Verzeichnis des Benutzers (bei Linux/Unix ist das meist `/home/user`, bei Windows meist `../Eigene Dateien`). Der dritte Button erzeugt einen Neuen Ordner in der dargestellten Verzeichnisstruktur. Der vierte und fünfte Button verändern die Anzeigedarstellung von der Ordnersicht auf die detaillierte Ansicht mit Angabe der Dateiengröße und Erstellungsdatum.

Abbildung 12 Zusatz Buttons beim Fenster Datei öffnen

- Natürlich kann man auch unter dem Punkt *„Dateiname“* den gewünschten Dateinamen manuell eingeben. Hier bitte auf korrekte Schreibweise achten, das das Programm sonst die Datei nicht finden kann.

- Unter dem Punkt *„Dateitypen“* kann der Benutzer zwischen dem Modus *Karten Dateien* (hier werden nur Ordner, *.JPG und *.KAP Dateien angezeigt) und dem Modus *Alle Dateien* wählen. Beim Modus Alle Dateien hat der Benutzer die freie Wahl der Auswahl, ich weise jedoch darauf hin, das das Programm nur mit echten *.JPEG und *.KAP Dateien arbeiten kann. Diese Möglichkeit ist nur für den Fall gedacht, das es Dateien gibt, die korrekte Daten enthalten, allerdings eine andere Dateiendung enthalten (Beispiel: eine *.JPEG Datei mit der Namen Vorsicht_jpg.dat).

- Über den Button *Öffnen* kann nun eine im Fenster ausgewählte Datei (welche sich auch über einen Doppelklick öffnen lässt) öffnen.

- Der Button *Abbrechen* verlässt dieses Dateiauswahlfenster ohne eine Datei zu laden.

7.2.1.2. Einstellungen

Über das Menü Datei können die Einstellungen der Software GPSMan erreicht werden. Die Einstellungen sind auch mit der Tastenkombination STRG-E zu erreichen. Die Maske der Einstellungen ist mithilfe von Registerkarten (TabbedPane) in einzelne Bereiche unterteilt.

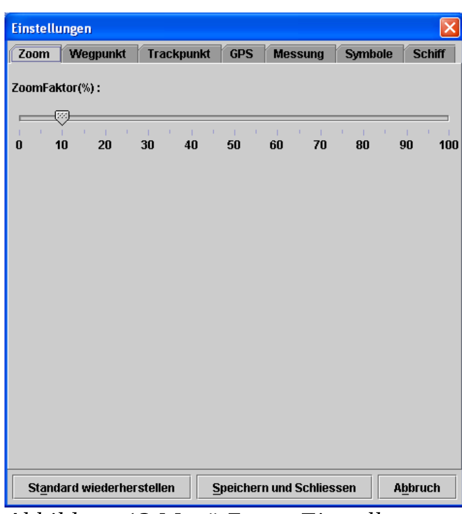


Abbildung 13 Menü Zoom, Einstellungen

- Zoom
- Wegpunkte
- Trackpunkte
- GPS
- Messungen
- Symbole

7.2.1.2.1. Register Zoom

Im Register Zoom kann der zum Zoomen der jeweiligen Seekarten voreingestellte Zoomfaktor von 10% geändert werden. Ein höherer Zoomfaktor bewirkt bei einem einmaligen Zoom eine grössere Beeinflussung des Ergebnisses. Der Zoomfaktor wird hier über einen Schieberegler eingestellt. Nach Betätigung des Buttons *Speichern und Schliessen* wird diese Einstellung im Programm gespeichert. Mit Betätigung des Buttons *Standard wiederherstellen* ist der Zoomfaktor wieder auf 10% eingestellt.

7.2.1.2.2. Register Wegpunkte

Das Register Wegpunkte zeigt dem Benutzer alle Möglichkeiten hinsichtlich der Darstellung der Symbole für die Wegpunkte. Als Standardsymbol ist hier das obere rechte Symbol gewählt. Jegliche neue Standardeinstellung für die Wegpunktsymbole kann der Benutzer nun hier treffen. Mit diesem Wegpunktsymbol werden dann alle neuen Wegpunkte gezeichnet. Auch hier muss zum Abschluss der Auswahl der Button *Speichern und Schliessen* betätigt werden, nur so werden die Einstellungen für das Programm übernommen.

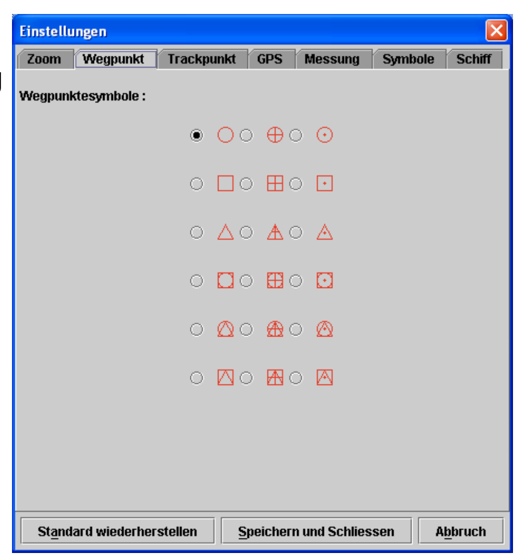


Abbildung 14 Register Wegpunkte, Einstellungen

7.2.1.2.3. Register Trackpunkte

Im Register Trackpunkte werden ähnlich wie bei den Wegpunkten dem Benutzer einige verschiedene Symbole zur Darstellung der Trackpunkte gezeigt. Auch hier ist das obere rechte Symbol das Standardsymbol. Durch Auswahl eines neuen Symbol kann der Benutzer hier die Standarddarstellung von Trackpunkten beeinflussen.

Zusätzlich wird hier aber noch die Farbe der Trackpunkte zur Auswahl gestellt. Ein Klick auf den Farbbutton führt den Benutzer in ein Auswahlfenster für Farben, wo er eine neue Auswahl der Farbdarstellung der Trackpunkte treffen kann. Im Unterschied zu den Wegpunkten (wo es nur die Farbdarstellung in Rot gibt) können hier bei den Trackpunkten auch andere Farben gewählt werden können. Die Standardauswahl der Farbdarstellung ist die Farbe Blau.

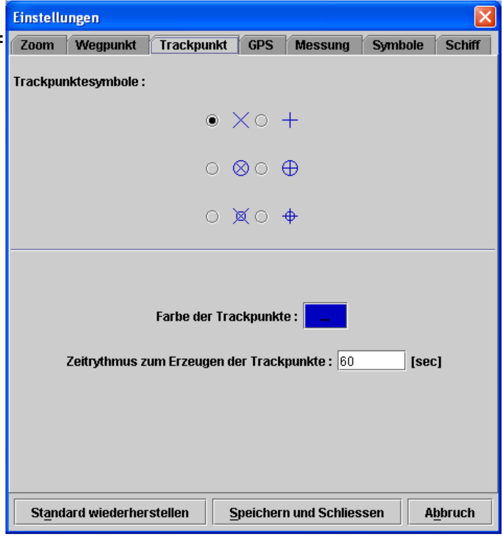


Abbildung 15 Register Trackpunkte, Einstellungen

Als letzter Konfigurationspunkt der Trackpunkte kann hier der Zeitrythmus zur Darstellung der Trackpunkte getroffen werden. Der Standardwert ist hier 60 Sekunden, was bedeutet, das im Falle vom Tracking (GPS oder Simulation) alle 60 Sekunden ein Trackpunkt auf der jeweiligen Seekarte erzeugt und dargestellt wird.

7.2.1.2.4. Register GPS

Im Register GPS findet der Benutzer Konfigurationsmöglichkeiten zum Tracking Modus.

Im ersten Anschnitt kann hier zwischen dem Empfangen von „echten“ GPS Daten aus einem GPS Gerät und dem Darstellen oder auch Simulieren von GPS Daten unterschieden werden. Nach dem Start des Tracking werden die Daten dann automatisch von der richtigen Quelle bezogen, im Falle des GPS (Online) Modus kommen die Daten dann von einem GPS Gerät und im Falle der Simulation (Offline) werden diese Daten im Programm berechnet und simuliert.

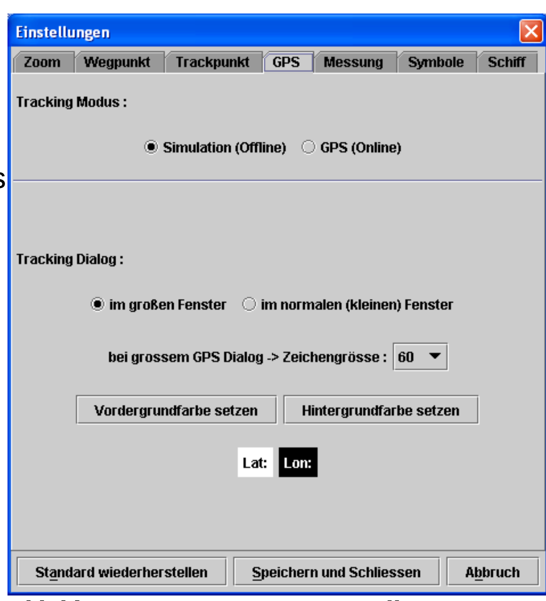


Abbildung 16 Register GPS, Einstellungen

Der zweite Abschnitt behandelt die Darstellung von GPS Daten im einem Dialogfenster. Der Benutzer kann sich hier für die Darstellung in einem grossen Fenster (siehe Kapitel Tracking Dialog) oder für die Darstellung in einem kleinen Fenster entscheiden. Zusätzlich kann der Benutzer die Grösse des grossen Dialogfenster noch massgeblich mit der Angabe, bzw. Veränderung der Zeichengrösse verändern. Standardauswahl ist hier die Zeichengrösse 60 (was passend für die Bildschirmauflösung 1024*768 gedacht ist).

Der letzte Abschnitt behandelt die Farbdarstellung im Tracking Dialog. Hier können die Vordergrundfarbe und die Hintergrundfarbe neu bestimmt werden. Die Standardauswahl ist hier die Kombination von Schwarz und Weiß.

7.2.1.2.5. Register Messungen

Im Register Messungen können Einstellungen zu den Mess- und Berechnungsverfahren getroffen werden.

Die Masseinheit für die Messung von Distanzen kann hier zwischen Seemeilen (Standard) und Kilometer gewählt werden. Die Auswahl mit Kilometer ist nicht für den Einsatz auf dem Wasser gedacht, wobei allerdings unkundigen Benutzer im Umgang mit der Simulation und der Masseinheit Kilometer ein Gefühl für die Entfernung gegeben werden kann.

Abbildung 17 Register Messungen, Einstellungen

7.2.1.2.6. Register Symbole

Im Register Symbole werden allgemeine Darstellungen von Symbolen aufgeführt.

Im ersten Abschnitt kann die Darstellung des Kurspfeiles (bei der Ermittlung von Kurswinkeln) in der Grösse seiner Darstellung beeinflusst werden.

Im zweiten und letzten Abschnitt kann der Benutzer entscheiden, ob er bei der Messung von Distanzen, die Masspfeile von oben oder von unter an die Messlinie herangeführt haben will.

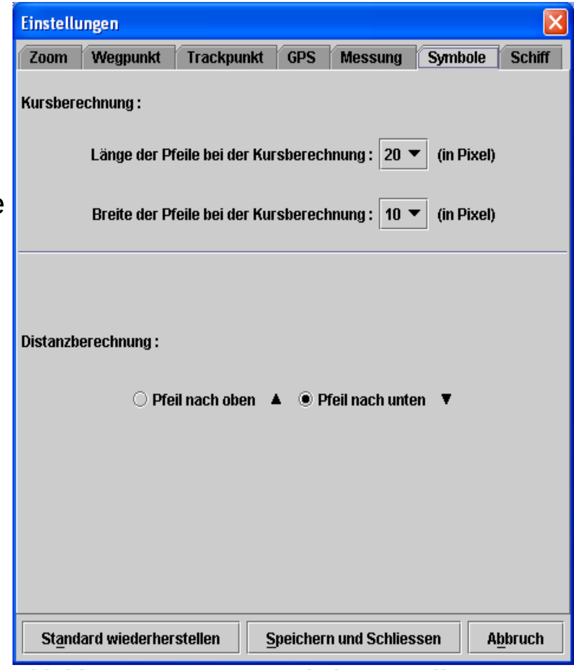


Abbildung 18 Register Symbole, Einstellungen

7.2.1.2.7. Register Schiff

Im Register Schiff kann der Benutzer eine Hilfslinie bei der Darstellung der aktuellen Schiffsposition aktivieren. Die Hilfslinie zeigt die Verlängerung des zur Zeit genutzten Kurswinkel, damit vereinfacht sich für den Schiffsführer die Ansicht der aktuellen Schiffsposition. Er sieht in Verlängerung seines Kurses evtl. Fahrtrichtungsänderungen früher und kann dementsprechend handeln. Die Länge der Hilfslinie kann hier definiert werden, ist im Standard aber 60 Pixel lang.

7.2.1.3. Beenden

Die Möglichkeit die Anwendung zu verlassen ist unter dem Menüpunkt *Datei / Beenden* gegeben. Der Menüpunkt *Beenden* ist zusätzlich über die Tastenkombination STRG-Q erreichbar.

7.2.2. Menü Logbuch

Der Benutzer kann mit einem sogenannten Logbuch arbeiten, welches man hier neu anlegen oder öffnen kann. (Die Programmierung dieser Menüpunktes ist nicht Teil meiner Diplomarbeit). In beiden Fällen öffnet sich derzeit ein Dialogfenster, welches auf das noch zu implementieren Logbuch hinweist. Ein ausführliche Programmierung und Beschreibung wird wahrscheinlich Teil einer zukünftigen Projektarbeit der Fachhochschule Trier sein. Der Menüpunkt *Logbuch öffnen* ist zusätzlich über die Tastenkombination STRG-L, der Menüpunkt *Neues Logbuch* über STRG-N erreichbar.



Abbildung 19 Menü Logbuch

7.2.3. Menü Karte

Unter dem Menü Karte findet der Benutzer weitere Menüpunkte die im Zusammenhang mit den dargestellten Karten sind.



Abbildung 20 Menü Karte

7.2.3.1. Kalibrieren

Der manuelle Kalibrierungsdialog kann hier aufgerufen werden, sofern man eine eingescannte Seekarte im JPEG Dateiformat manuell kalibrieren möchte (welches bei der weiteren Benutzung der Karte Sinn macht). Die Benutzung der Maske wird mit

unterstützenden Hilfetexten (in Rot) in der untersten Zeile dieses Fenster

Abbildung 21 manuelle Kalibrierungsmaske

erleichtert. Die drei Punkte, die kalibriert werden müssen, werden hier mit Ihren Pixelkoordinaten und Längen- und Breitengraden dargestellt. Des weiteren werden Auswahlmöglichkeiten für die Ellipsoide und die Projektionen dargestellt. Diese Auswahlmöglichkeiten sind jedoch insofern deaktiviert, das keine andere Auswahlmöglichkeit als WGS84 und Mercator möglich sind. Für eine spätere Erweiterung wurden jedoch die Möglichkeit dazu mit eingebaut. Im unteren Bereich sind die drei Bestätigungsbuttons (Berechnen, Berechnen und Speichern, sowie Abbruch) welche folgende Funktionalitäten bereitstellen:

- Berechnen

  Es werden die Daten nur berechnet, d.h. die Karte wird nach den vorliegenden Daten kalibriert. Eine Speicherung der Kalibrierungsdaten erfolgt aber nicht.

- Berechnen und Speichern

  Es werden die Daten berechnet, d.h die Karte wird nach den vorliegenden Daten kalibriert und eine Speicherung der Kalibrierungsdaten erfolgt. Es wird dann eine Datei mit dem Namen *„<Name der Grafikdatei.typ>.cal“* erzeugt und in das gleiche Verzeichnis wie die Grafikdatei gespeichert.

- Abbruch

  Die Kalibrierungsmaske wird ohne Kalibrierung der Karte verlassen. Danach wird der Mauszeiger wieder auf den normalen Status umgestellt, damit der Benutzer erkennt, das er den Kalibrierungsmodus verlassen hat.

Die Eingabe der Daten erfolgt manuell, durch Eintragen der Pixelkoordinaten per Hand oder man klickt mit der Maus (die jetzt mit einem Kreuz-Cursor diesen Modus anzeigt) in die Karte hinein. GPSMan überträgt dann die entsprechenden Pixelkoordinaten automatisch in die Kalibrierungsmaske. Nun muss der Benutzer die entsprechenden Längen- und Breitengrad manuell nachtragen und eine Kalibrierung kann

durchgeführt werden. Die Bearbeitung ist immer nur für einen der drei Punkte möglich, so wird verhindert, das versehentlich falsche Werte übertragen werden. Beim Laden einer bereits kalibrierten Karte im JPG-Format wird die Kalibrierungsdatei automatisch geladen (sofern diese dann im gleichen Verzeichnis wie die Seekarte ist). Beim Aufruf der Kalibrierungsmaske können die Werte dann eingesehen werden. Der manuelle Kalibrierungsmodus kann mit der Tastenkombination STRG-K sofort erreicht werden.

7.2.3.2. Daten zur KAP-Datei

Zu eingelesenen Seekarten im KAP-Format kann man die internen Daten in einem weiteren Fenster ansehen. Hier werden Daten über den Namen der Seekarte und deren derzeitiger Speicherort aufgeführt. Des weiteren kann man hier Daten wie Maßstab, Nummer der Karte, Breite und Höhe der Karte in Pixel, sowie Umrechungsfaktoren von Pixel in inch erfahren. Im unteren Teil sind dann auch die Kalibrierungsdaten der Seekarte ersichtlich. Die Anzeige der Karteninformationen kann mit der Tastenkombination STRG-D sofort aktiviert werden.



*Abbildung 22 Informationen zur *.KAP Seekartendatei*

7.2.4. Menü GPS

Der Onlinemodus des GPS Gerätes kann eingeschaltet werden und die eintreffenden Daten können in einem GPS Dialogfenster verfolgt werden. Die Besonderheit dieses Menüpunktes ist so darzustellen, das es sich hier um ein Einschalten – Ausschalten Modus handelt. Diese Funktion ist auch in der Toolbar zu finden. (Die weitere Programmierung dieser Funktionalität ist nicht Teil meiner Diplomarbeit).

7.2.4.1. Tracking

Das Tracking oder der GPS Modus schaltet den Einlesemodus der GPS Daten von einem jeweiligen GPS Gerät ein oder startet die Simulation. Die Übertragung der GPS Protokolle von einem GPS Gerät sind nicht Teil dieser Arbeit. Beim Einschalten wechselt das Icon des Buttons GPS Modus einschalten seine Darstellung, aus der Erdkugel mit einem roten Kreuz (Modus ausgeschaltet) wird eine Erdkugel mit einem grünen Haken (Modus eingeschaltet). Dadurch ist es für den Benutzer ersichtlich das der GPS Modus aktiv ist.



Abbildung 23 Menü GPS

- **Simulation**

Nach dem Starten des Tracking werden im Falle der Simulation die Bewegungsdaten eines Schiffes vom Programm berechnet. Das Schiff wird simuliert. Hierzu sind einige Startparameter zur Simulation notwendig. Beim Starten der Simulation erscheint daher ein Dialog in dem diese

Startparameter eingestellt werden müssen.

Die Startparameter sind folgende

- der Startpunkt des Schiffes durch Angabe einer Position (hier kann auch mit der Maus diese Position aus der jeweiligen Seekarte durch einen Linksklick übernommen werden. Der Fadenkreuz-Cursor zeigt dem Benutzer dies an).

- Die Geschwindigkeit (in Knoten) und die Richtung in die sich das Schiff bewegt.

- Die Abweichung, die durch Gegenwind und/oder Gegenströmung entstehen kann (Geschwindigkeit und Winkel)

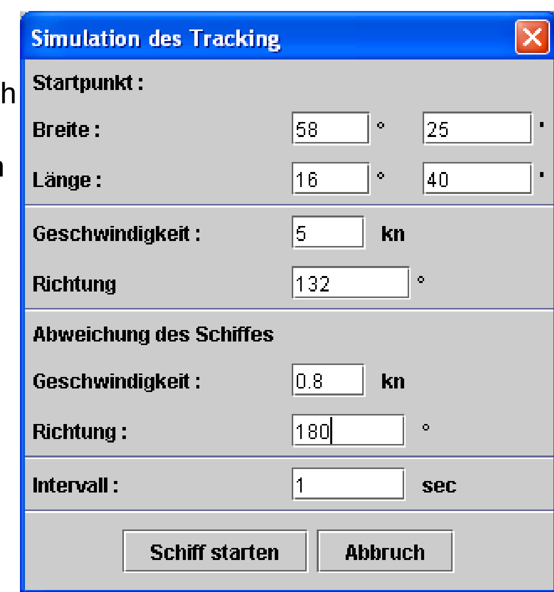


Abbildung 24 Simulatoin des Tracking

- Das Zeitintervall, nach dem das Programm eine neue Position des Schiffes berechnen soll.

Mit Betätigen des Buttons Schiff starten wird die Simulation gestartet.

- **Steuerung Simulation**

Nach dem Starten der Simulation wird ein sogenanntes Steuerungsdialog in der oberen rechten Ecke des Bildschirm dargestellt. Mithilfe dieser Steuerungshilfe kann der Benutzer während der Simulation die verschiednen Parameter der Simulation beeinflussen.

- Geschwindigkeit und Kurs des Schiffes

- Geschwindigkeit und Winkel der Abweichung

Nach Betätigen des Buttons Änderung OK werden diese Daten an die Simulation weitergeleitet und beim nächsten Zeitintervall eingestellt und genutzt.

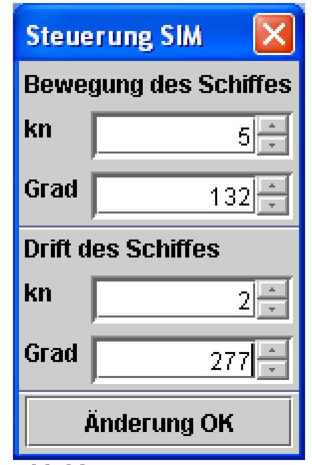


Abbildung 25 Steuerung SIM

Der Steuerungsdialog kann nicht entfernt werden, er verschwindet erst mit Beenden der Simulation (Betätigen des Buttons Tracking oder im Menü unter Tracking). Somit geht dem Benutzer die Möglichkeit zur Beeinflussung der Simulation nie verloren.

7.2.4.2. Tracking Dialog

GPS Dialog
Lat: 12°12,12'
Lon: 53°14,45'
KüG: 180,4°
FüG: 10,6 kn
Dat: 04.01.04
BZ: 19:30:34

Abbildung 26 Große GPS Dialog

Wird der GPS Dialog eingeschaltet, so ist vorgesehen, das man während der Benutzung des GPS Gerätes, sprich während der ständigen Datenaustausch des GPS Gerätes mit der Software, visuell diese Daten in dem GPS Dialog mitverfolgen kann. Diese Funktion wird auch in der Simulation verwendet. Hierzu werden in dem GPS Dialog Fenster die Daten der aktuellen Breiten (Lat)- und Längengrad (Lon) angezeigt. Des weiteren wird hier die Fahrt über Grund (FüG) in Knoten (kn) und der Kurs über Grund (KüG) angezeigt. Als weitere wichtige Daten sind das Datum (Dat) und die aktuelle Bordzeit (BZ) noch angegeben. Dieses Fenster ist bewusst gross dimensioniert, damit der Benutzer bei der Fahrt mit einem Segelschiff ohne große Probleme die Kerndaten eines GPS System auch aus einiger Ferne (vom Schiffssteuer z.B.) erkennen kann. Die Größe ist über die Einstellungen veränderbar. Außer der großen Variante, kann der Benutzer auch in den Einstellungen auf die kleine Maske umschalten (siehe *Optionen / Einstellungen*), die in der normalen Größe der anderen Dialogmasken dargestellt wird. Der Menüpunkt *GPS Dialog* ist zusätzlich über die Tastenkombination STRG-G erreichbar.

7.2.4.3. Aktuelle Schiffsposition



Abbildung 27 Schiffposition

Der letzte Menüpunkt in dem Menü Karte lässt dem Benutzer die Möglichkeit offen die derzeitige Schiffsposition grafisch auf der Seekarte darzustellen. Die Position wird durch ein Fähnchen mit der Inschrift „Ship“ dargestellt, zusätzlich informiert ein Kurspfeil über den derzeitigen Kurs des Schiffes. Die Spitze des Kurspfeiles markiert die aktuelle Position des Schiffes.

Dieser Punkt macht aber nur dann Sinn, wenn das GPS Gerät eingeschaltet ist und damit unentwegt Daten sendet oder der Simulationsmodus eingeschaltet ist und somit ein eingeschaltetes GPS Gerät simuliert. Die Darstellung der Schiffsposition kann mit der Tastenkombination STRG-P sofort aktiviert werden.

7.2.5. Menü Punkte

Im Punkte-Menü hat der Benutzer die Möglichkeit sich für die Erstellung von verschiedener Punktesystemen in der Karte zu entscheiden. Hier gibt es die Möglichkeit den so genannten Wegpunkte Modus in einem Einschalten – Ausschalten Modus zu setzen (und damit Wegpunkte in der Karte zu erzeugen). Des weiteren können Sie Wegpunkte unter dem Menüpunkt *Wegpunkte verwalten* so genannten Routen zugeordnet werden. Eine ähnliche Vorgehensweise findet sich in der Handhabung der Trackpunkte und der Tracks. Natürlich können alle Punktsysteme in dafür vorgesehenen Dateien gesichert

und auch wieder geladen werden. Erzeugt werden hier reine XML-Dateien mit der Dateiendung *.tp (für trackpoint).

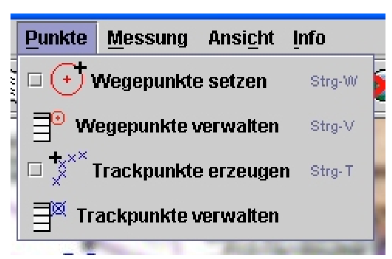


Abbildung 28 Menü Punkte

7.2.5.1. Wegpunkte erzeugen

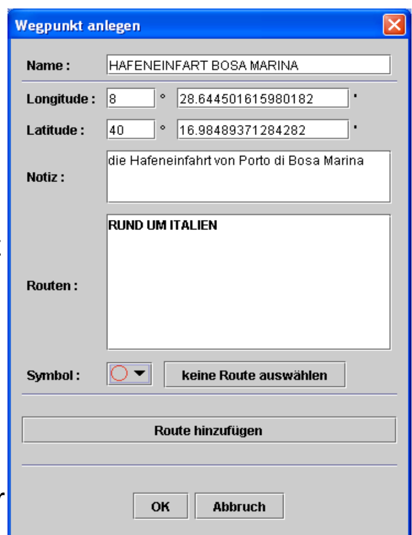


Abbildung 29 Wegpunkte erzeugen

Aktiviert man den Modus Wegpunkte setzen, so ist der Benutzer nun in der Lage in der jeweiligen Seekarte Wegpunkte zu erzeugen. Erkennen kann das der Benutzer an der Tatsache, das der Button Wegpunkte setzen grau hinterlegt ist (also aktiviert ist), der Menüpunkt Wegpunkte setzen ein Häkchen hat und das der Cursor von der Pfeildarstellung in das Fadenkreuz gewechselt ist. Nun ist das System bereit einen Wegepunkt zu markieren. Klickt der Benutzer nun mit der linken Maustaste, so erscheint der Wegepunktdialog, indem der Benutzer den erstellten Wegpunkte näher definieren kann. Alle Felder sind editierbar, sodass der Benutzer den erstellten Wegpunkte auch noch manuell durch Änderung der Breiten- und Längengrade anpassen kann. Des weiteren können Notizen zu dem Wegepunkt eingetragen werden. Sollte der Benutzer einen Wegpunktenamen vergessen, so erstellt das System einen automatischen Namen (indem es die verschiedenen Wegpunkte durchnummeriert W1, W2, W3, usw.). Bei der Eintragung des Wegepunktnamens ist zu beachten, das hier nicht alle Zeichen erlaubt sind und das nur in Großbuchstaben geschrieben wird. Fehler seitens des Benutzers sind hier aber nicht möglich, da die Software nur die zulässigen Zeichen darstellt und alle Zeichen automatisch in Großbuchstaben umwandelt. Diese Einschränkungen sind bei dem Eintrag der Notiz nicht vorhanden, hier kann der Benutzer alle Zeichen verwenden. Zusätzlich ist in dem Dialogfenster zum Erstellen von Wegpunkte auch noch die Möglichkeit zum Zuordnen von Routen mit vorgesehen. Unterhalb des Notizfeldes ist eine Tabelle mit allen derzeit im System vorhandenen Routen eingefügt. Das Auswählen der Routen funktioniert durch einfaches Anklicken der entsprechenden Route. Möchte der Benutzer mehrere Route auswählen, so kann man dies mit Drücken der STRG Taste und gleichzeitigem Anklicken der entsprechenden Routen bewerkstelligen. Diese Mehrfachauswahl ist in vielen Betriebssystemen Standard. Sollte noch keine Route definiert worden sein, so kann der Benutzer eine neue Route mit dem Button *Route hinzufügen* erzeugen (über die Verfahrensweise hierzu später im Kapitel *Route erzeugen* mehr). Jeder Wegepunkt wird in der Seekarte mit einem Symbol und seinem Namen dargestellt, daher hier noch die Möglichkeit eine entsprechendes Symbol aus dem Symbolfundus des System auszuwählen. Betätigt der Benutzer nun nach Eintragung und Auswahl aller Daten den *OK* Button, so werden diese Daten in der Software gesetzt. Das Dialogfenster verschwindet und der Wegepunkt wird auf der Seekarte dargestellt. (Vorsicht, ist die Seekarte nicht kalibriert, so können die Wegpunkte NICHT dargestellt werden). Betätigt der Benutzer jedoch den *Abbruch* Button, so gehen alle Daten verloren und es wird kein zusätzlicher Wegepunkt auf der Seekarte dargestellt.

7.2.5.2. Wegpunkte verwalten

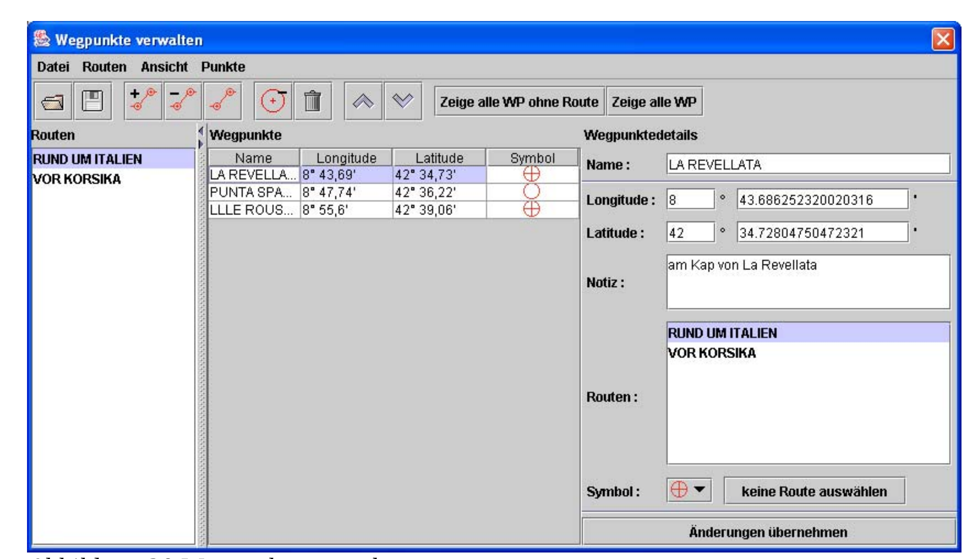


Abbildung 30 Wegpunkte verwalten

Sind nun eine bestimmte Menge von Wegpunkten erzeugt, so hat der Benutzer über den Menüpunkt *Wegpunkte verwalten*, die Möglichkeit die Wegpunkte zu Routen zusammen zufassen und ggf. Änderungen einzufügen. Das Dialogfenster *Wegpunkte verwalten* ist in der Benutzung eines der komfortabelsten, aber auch eins der komplexesten Masken in der Software GPSMan. Durch die Vielfältigkeit der Möglichkeiten ist die Komplexität der Maske zu erklären. Daher sollte vor der Benutzung des Menüpunktes *Wegpunkte verwalten* dieses Kapitel durchgelesen werden. Der Menüpunkte *Wegpunkte verwalten* ist zusätzlich über die Tastenkombination STRG-V zu erreichen.

- Im linken Bereich ist die Liste der Routen zu erkennen. Sind in diesem Fenster keine Einträge zu sehen, so hat das System derzeit keine Route definiert (man kann Sie über den Menüpunkt und Button Route hinzufügen neu erstellen). Durch Anklicken einer Route wird diese Route aktiviert und im mittleren Bereich wird die Anzeige der Wegpunkte auf diese Route gefiltert. D.h. es werden nur diejenigen Wegpunkte angezeigt, die dieser aktivierten Route zugeordnet wurden. Im oberen Beispiel ist die Route RUND UM ITALIEN aktiviert und es werden die dieser Route zugeordneten Wegpunkte gezeigt. Die aktivierte Route wird blau in der Darstellung hinterlegt. Eine aktivierte Route kann mit dem Menüpunkt *Route löschen* oder mit dem Button *Route löschen* aus dem System entfernt werden. Alle ihr zugeordneten Wegpunkte sind noch vorhanden, allerdings ohne die Zuordnung der gelöschten Route. Zusätzlich kann eine aktivierte Route auch über den Menüpunkt *Route ändern* oder den Button *Route ändern* in ihrer Definition verändert werden. Eine Beschreibung dieser Maske im Kapitel Route hinzufügen, da die Masken zum Erstellen und Ändern von Routen identisch sind. Der linke Bereich kann in seiner Größe gegenüber dem mittleren Bereich verändert werden. Hierzu die Abtrennung der beiden Bereiche mit der linken Maustaste anklicken, die linke Maustaste gedrückt halten und den so genannten SplitPane verschieben in die gewünschte Richtung. Der rechte Bereich ist in seiner Größenordnung fest vorgegeben und kann nicht verändert werden.

- Im mittleren Bereich werden die Wegpunkte als Liste oder Tabelle dargestellt. Die Position der Wegepunkte in der Route kann mit den Pfeilen in der Buttonleiste verändert werden. Man aktiviert eine Wegepunkt durch

Anklicken mit der linken Maustaste und verschiebt diesen Wegepunkt in seiner Position mit den Pfeilbuttons nach oben oder unten. Eine Mehrfachauswahl zum Verschieben von Punkten wird nicht unterstützt. Ein aktivierter Wegepunkt wird im rechten Bereich, dem so genannte Wegepunktdetailbereich mit all seinen Daten angezeigt. Ein aktivierter Wegepunkt kann mit dem Menüpunkte *Wegepunkt löschen* oder mit dem Button *Wegepunkt löschen* aus dem System gelöscht werden. Alle Routen entfernen dann diese Wegepunkt aus ihren Zuordnungen und der Wegepunkt wird komplett aus dem System entfernt. Die Punkte *Wegpunkte löschen* entsprechen nicht der Entfernung der Zuordnung in den Routen, sondern hier werden die Punkte physikalisch aus dem System entfernt. Für die Entfernung der Zuordnung zu einer Route müssen Änderungen im Wegepunkt im rechten Bereich, dem Wegepunktdetailbereich gemacht werden. Allerdings ist im mittleren Bereich die Änderung des Wegepunktsnamens und die Zuordnung des Symbols für den Wegepunkt möglich. Die Anzeige der Breiten- und Längengrade ist im mittleren Bereich nicht möglich. Zusätzlich zu der Möglichkeit einzelne Wegepunkt zu löschen, gibt der Menüpunkt *Alle Wegpunkte löschen* oder / und der Button *Alle Wegpunkte löschen* dem Benutzer die Gelegenheit alle Wegepunkt in einem Zuge physikalisch aus dem System zu entfernen. Als letztes zu nennen noch die Möglichkeiten sich alle Wegepunkt ohne Routenzuordnung anzeigen zu lassen über den Menüpunkte *Ansicht / Zeige alle Wegpunkte ohne Routenzuordnung* oder über den Button *Zeige alle WP ohne Route*, sowie die Möglichkeit sich alle Wegpunkte über den Menüpunkt *Zeige alle Wegpunkte* und über den Button *Zeige alle WP* im mittleren Bereich darstellen zu lassen.

- Der rechte Bereich, der so genannte Wegepunktdetailbereich zeigt alle Daten bezüglich eines Wegpunktes (des aktivierten Wegpunktes). Alle Daten können hier geändert werden. Auch die Zuordnung zu den einzelnen Routen wird hier vorgenommen. Hierzu ist hier auch eine komplette Liste der Routen vorgesehen. Der Wegepunktdetailbereich ist identisch mit der Dialogmaske zum Erstellen von Wegpunkten, bis auf den Button Änderungen übernehmen. Dieser Button muss betätigt werden, damit alle getätigten Änderungen an den Daten eines Wegpunktes erhalten bleiben und im mittleren Bereich ebenso angezeigt werden. Vergisst der Benutzer diesen Button nach Änderung von Daten (Hinzufügen von Routenzuordungen) zu betätigen, gehen alle Änderungen verloren (dieser Sachverhalt kann unter Umständen auch gewollt sein).

7.2.5.2.1. Menüpunkt Datei

Das Dialogfenster Wegpunkte verwalten besitzt eine eigene Menüleiste die mit dem Menüpunkt Datei beginnt. Hier kann der Benutzer *Wegpunkte und Routen laden* und/oder *Wegepunkt und Routen sichern*. Die Wegepunktsysteme werden in XML-Dateien mit der Dateiendung *.wp (waypoint) gesichert. Genau wie bei den Trackpunkten können diese Dateien mit jedem Browser eingesehen werden und auch ggf. editiert werden. Als letzter Menüpunkt ist hier das *Beenden* der Menümaske *Wegpunkte verwalten* vorgesehen. Die

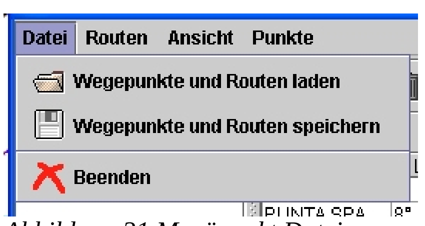


Abbildung 31 Menüpunkt Datei, Wegepunkt verwalten

Dialogfenster zum Laden und Sichern von Wegepunktsystemen sehen sich sehr ähnlich und sind in ihrer Benutzung fast identisch mit der Maske zum Laden ein Seekarte (siehe Kapitel *Karte / Bild öffnen*)

7.2.5.2.2. Menüpunkt Routen

In dem Menüpunkt Routen findet der Benutzer alle Möglichkeiten zum Erzeugen, Löschen und Ändern von Routen. Über den Menüpunkt Route hinzufügen kann man neuen Routen definieren, mit Route löschen kann man aktivierte Routen (man aktiviert Routen durch Anklicken mit der linken Maustaste) im *Wegpunkte verwalten* Fenster löschen. Mit dem Menüpunkt Routen ändern kann der Benutzer bestehende Definitionen von Routen und deren Daten ändern. Für den Menüpunkte Route hinzufügen und den Menüpunkt Route ändern gibt es eine zusätzlich Maske, die es dem Benutzer ermöglicht die Daten zu eine Route zu definieren und ggf. später Änderungen daran vorzunehmen.

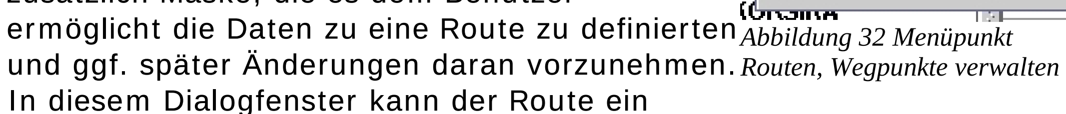


Abbildung 32 Menüpunkt Routen, Wegpunkte verwalten

In diesem Dialogfenster kann der Route ein Namen vergeben werden, hier gelten allerdings die gleichen Restriktionen wie bei den Wegepunktnamen. Das Eingabefeld ist aber nur in der Lage gültige Zeichen entgegen zunehmen und wandelt diese automatisch in Großbuchstaben um. Genau wie bei den Wegpunkten, so können auch hier bei den Route freie Notizen hinterlegt werden. Die Eintragung in das Notizfeld sind frei von jeglichen Restriktionen. Des weiteren hat jede Route eine eigene Farbe (die Route werden später in der Seekarte mit den jeweiligen Farben dargestellt und dadurch unterscheidbarer gemacht). Zur Auswahl der Farben hat der Benutzer die Möglichkeit sich nach bereits vorbereiteten Muster zu richten oder er entwirft seine Routenfarbe frei nach dem *HSB* oder / und *RGB* Farbmodell. Ein Vorschaufenster zeigt dem Benutzer die jeweilig definierte Farbe an. Mit Betätigen des *OK* Buttons werden die Daten gespeichert und die jeweilige Route wird entsprechend ihrer Wegpunkte in der Seekarte dargestellt.

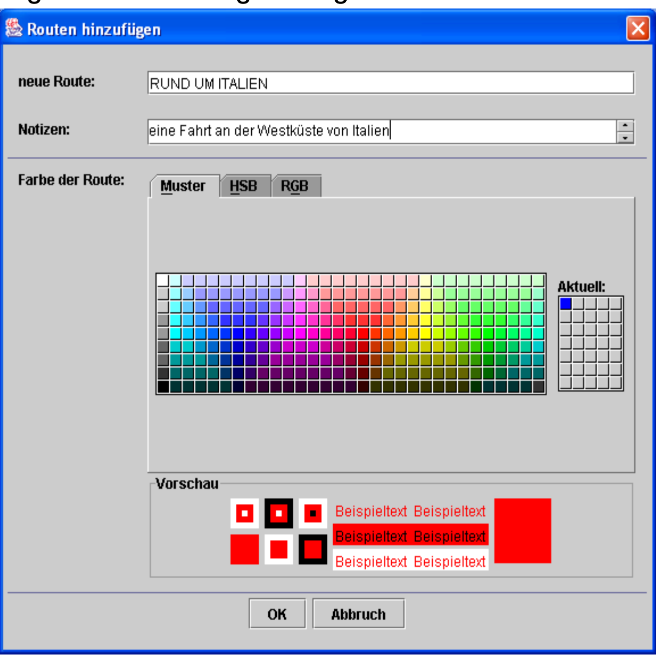


Abbildung 33 Route hinzufügen / ändern

7.2.5.2.3. Menüpunkt Ansicht

Der Menüpunkt Ansicht beherbergt die beiden Menüpunkte *Zeige alle Wegepunkt ohne Routenzuordnung* und *Zeige alle Wegpunkte*. Über diese

Menüpunkte wird die Darstellung der Wegepunkt in dem mittleren Bereich beeinflusst. Der Menüpunkt *Zeige alle Wegepunkt ohne Routenzuordnung* zeigt dem Benutzer alle Wegepunkt im mittleren Bereich an, die über keinerlei Zuordnung zu einer Route verfügen. Mit dem Menüpunkt *Zeige alle Wegpunkte* werden ungeachtet der Routenzuordnung alle Wegepunkt im mittleren Bereich angezeigt.

Abbildung 34 Menüpunkt Ansicht, Wegpunkte verwalten

7.2.5.2.4. Menüpunkt Punkte

Als letzter Menüpunkt ist der Menüpunkt *Punkte* im Dialogfenster *Wegpunkte verwalten* noch zu erwähnen. Er besitzt den Menüpunkte *Wegepunkt nach oben verschieben*, mit dem es dem Benutzer möglich ist aktivierte Wegpunkte in ihrer Reihenfolge innerhalb einer Routenzuordnung zu verändern. Die gleiche Funktionalität bietet der Menüpunkt *Wegepunkt nach unten verschieben*, wohlweislich aber in die andere Richtung. Der Menüpunkt *Wegepunkt löschen* löscht den derzeitig aktiven Wegepunkt. Mit dem Menüpunkt *Alle Wegpunkte löschen* werden alle Wegepunkt gelöscht.

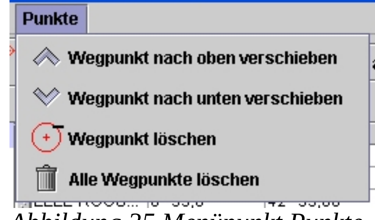


Abbildung 35 Menüpunkt Punkte, Wegepunkt verwalten

7.2.5.2.5. Buttonleiste des Dialogfenster Wegpunkte verwalten

Die Buttonleiste (Toolbar) des Dialogfenster Wegpunkte verwalten gibt dem Benutzer die Möglichkeit auf die verschiedene Funktionen schneller



Abbildung 36 Buttonleiste, Wegpunkte verwalten

zuzugreifen, letztendlich bietet Sie aber den gleichen Leistungsumfang, wie die in den einzelnen Menüpunkten enthaltenen Funktionalitäten. In der Reihenfolge von links nach rechts sind dies die Funktionen:

- Wegepunktdateien laden und öffnen
- Wegepunktdateien sichern
- Routen hinzufügen
- Routen löschen
- Routen ändern
- Wegpunkt löschen
- alle Wegpunkte löschen
-
- Wegepunkt nach oben verschieben

- Wegepunkt nach unten verschieben
- Zeige alle Wegpunkte ohne Routenzuordnung
- Zeige alle Wegpunkte

7.2.5.3. Menü Trackpunkte erzeugen und darstellen

Werden innerhalb der Software Bewegungsdaten erzeugt, so z.B. wenn ein GPS Gerät angeschlossen ist und die Simulation eingeschaltet ist, so werden diese aktuellen Bewegungsdaten auch innerhalb der Software gespeichert. Mit dem Menüpunkt Trackpunkte anzeigen kann man nun diese Bewegungsdaten visuell auf der Karte als Trackpunkte darstellen lassen. Jeder der Trackpunkte kann in einer Liste nachbearbeitet werden, so das jeder Trackpunkt über folgende Parameter verfügt:

- Position Breitengrad
- Position Längengrad
- Namen
- Notizen oder Logbucheintrag
- Symbol
- Geschwindigkeit
- Kurs
- Farbe

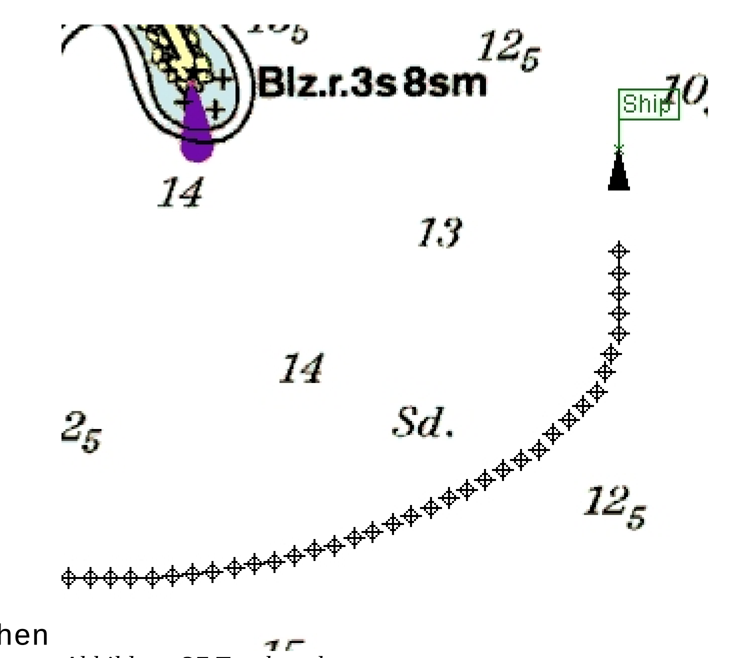


Abbildung 37 Trackpunkte

Wie in der Abbildung 30 zu ersehen ist, werden diese Trackpunkt visuell auf der Seekarte dargestellt.

7.2.5.4. Menü Trackpunkte verwalten

Um nun die Trackpunkte auf nachbearbeiten bzw. verwalten zu können, ist ein Trackpunkte Verwaltungsmodus dem Programm hinzugefügt worden.

Der Verwaltungsmodus der Trackpunkte ist ähnlich aufgebaut wie der Verwaltungsmodus der Wegpunkte. Ebenso ist hier die Möglichkeit gegeben, Tracks in eine Datei zu speichern und auch wieder Dateien dieser Art zu laden. Eine Trackdatei ist eine XML-Datei mit der Dateiendung *.tp, d.h. solche Dateien können mit jedem Browser eingesehen oder bearbeitet werden.

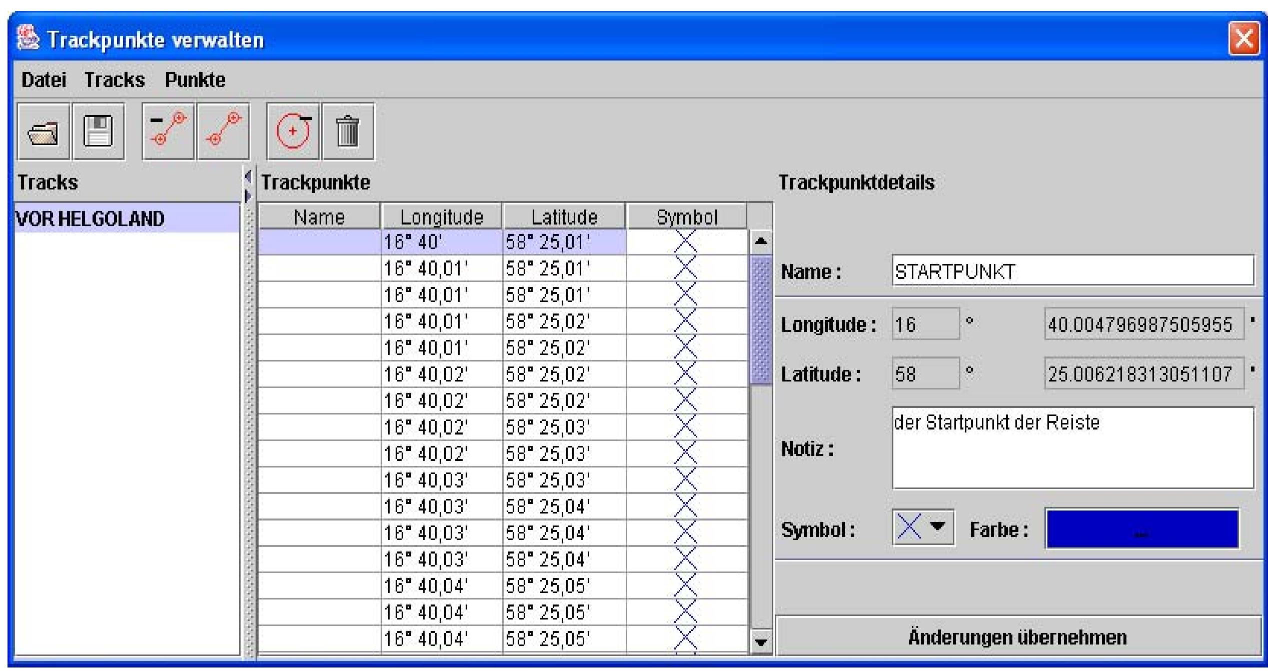


Abbildung 38 Trackpunkte verwalten

Im linken Bereich sieht der Benutzer die einzelnen Tracks, allerdings mit dem Unterschied zu den Wegpunkten das er hier nur bereits bestehende Tracks ändern bzw. anpassen kann. Tracks werden erzeugt durch das Starten der Trackpunkte erzeugen Funktion. Man kann aber die Tracks ändern über das Menü *Tracks / Tracks ändern* oder über den Button *Tracks ändern*. In dem nun gezeigten Dialog kann der Benutzer den Namen und die Notiz des Tracks ändern.

Im mittleren Bereich werden alle Trackpunkt des jeweiligen Tracks (bitte den jeweiligen Track zur Darstellung seiner Punkte anklicken und aktivieren) in einer Tabelle aufgelistet. Hier kann der Benutzer durch Aktivieren der einzelnen Trackpunkte durch Anklicken mit der linken Maustaste, diesen Trackpunkt im rechten Bereich weiter bearbeiten. Im mittleren Bereich können der Name und das Symbol des jeweilig aktivierten Trackpunktes verändert werden. Ähnlich wie bei der Verwaltung der Wegpunkte können hier auch aktivierte Trackpunkte durch das Menü Punkte / Trackpunkt löschen oder durch den entsprechenden Button gelöscht werden. Die Möglichkeit alle Trackpunkte (und damit auchdie Darstellung der Punkte in der jeweiligen Seekarte) zu löschen ist auch vorhanden.

Im rechten Bereich, dem sogenannte Trackpunkt Detailbereich können alle Daten zu einem Trackpunkt eingesehen werden. Verändert werden dürfen hier der Name, die Notiz, das Symbol und die Farbe des Trackpunktes. Nicht verändert werden können die Position des Trackpunktes. Zur Speicherung der Daten bitte immer den Button *Änderungen übernehmen* betätigen, da ansonsten die Daten beim Anwählen eines anderen Trackpunktes unwiderruflich verloren gehen.

7.2.6. Menü Messung

Das Menü Messung ist zum Auflisten der bereitgestellten Messungen und Berechnungen erstellt worden. Als erste Möglichkeit können Entfernungen

zwischen zwei Punkten bestimmt werden. Hierzu wählt man den Punkt *Entfernung messen* aus. Der Cursor wechselt nun wieder von der Standard Pfeil Ansicht in die Fadenkreuz Ansicht und man kann zwei Punkte auf der jeweiligen Seekarte anwählen. Zwischen diesen beiden Punkte wird nun die Messung anhand einer Linie mit zwei Endpunkten als Pfeile dargestellt. Die Form der Pfeile können unter *Optionen / Einstellungen* geändert werden. Das Ergebnis der Messung kann dem Messerergebnisfenster entnommen werden, welches automatisch nach jeder neuen Messung erscheint. Die Messlinie bleibt nun in der Seekarte vorhanden, bis man eine neuen Distanzmessung vornimmt oder über den Menüpunkt *eingetragene Entfernungsmessung löschen* die Linie wieder von der Seekarte entfernt. Die Entfernungsmessung ist zusätzlich über die Tastenkombination STRG-M zu erreichen. Zusätzlich zu der Entfernungsmessung können aber auch Berechnungen von Kursinformationen erfolgen. Hierzu wählt man den Menüpunkt Kurs berechnen an und kann wieder mit den Fadenkreuz zwei Punkte auf der Seekarte markieren. Nun berechnet die Software den Winkel zwischen diesen beiden Punkten und zeichnet einen Kurspfeil in die Seekarte ein. Das Ergebnis der Berechnung wird wieder in das Ergebnisfenster eingetragen. Die Verhaltensweise des Kurspfeil ist die gleiche wie bei der Messlinie, entweder man berechnet einen neuen Kurs, womit die alte Darstellung verschwindet und die neue Darstellung erscheint oder man löscht den Kurspfeil mit dem Menüpunkt *eingetragene Kursberechnungen löschen*. Der Kurspfeil ist in seiner Größe über *Optionen / Einstellungen* konfigurierbar.

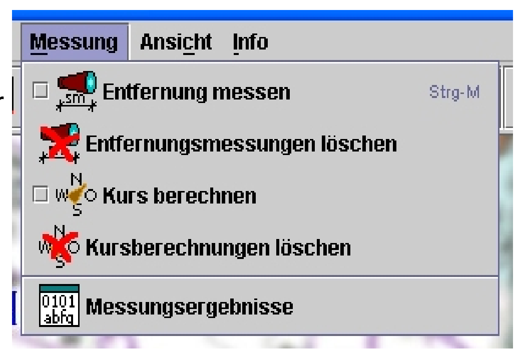


Abbildung 39 Menü Messung

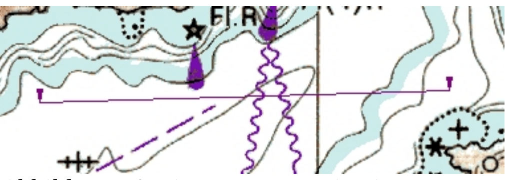


Abbildung 40 Distanzmessung in einer Seekarte



Abbildung 41 Messergebnis Distanzmessung

Abbildung 42 Kurspfeil

7.2.6.1. Menü Messungsergebnisse

Das bereits im Kapitel Menü *Messung* erwähnte Messergebnisfenster kann natürlich ein- und auch wieder ausgeblendet werden. Diese Funktionalität kann über den Menüpunkt *Messungsergebnisse* im Menü *Messung* erreicht werden. Das Ergebnisfenster besitzt die Eigenschaft bei jedem neuen Mess- oder Berechnungsverfahren automatisch wieder mit den jeweiligen Ergebnissen auf dem Bildschirm zu erscheinen. Wird keine neue Messung oder Berechnung getätigt, bleiben die alten Ergebnisse bis zum Beenden der Software über das Messergebnisfenster abrufbar. Ebenso wie Zeichnung der Messungen in den jeweiligen Seekarten. Messungen und Berechnungen besitzen kein Verfahren um in eine Datei gesichert zu werden. Löschen Sie Messungen und Berechnungen, so sind deren Ergebnisse auch automatisch aus dem Messergebnisfenster gelöscht.



Abbildung 43 Messergebnis Kursberechnung

7.2.7. Menü Ansicht

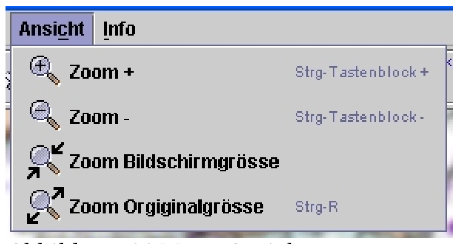


Abbildung 44 Menü Ansicht

Karten, die einmal geladen worden sind können mithilfe der Zoomfunktionen optisch verändert werden. Hier können die Images vergrößert, verkleinert und / oder wieder in der Ursprungsgröße dargestellt werden. Der grundsätzliche Zoomfaktor ist mit dem Wert 10% in der Grundeinstellung der Software vorgesehen. Änderungen an dem Zoomfaktor können aber in den Einstellungen unter dem Menüpunkt *Optionen / Einstellungen* vorgenommen werden. Mit dem Menüpunkt Zoom+ können Vergrößerungen der Kartendarstellung, mit dem Menüpunkt Zoom- Verkleinerungen erzielt werden. Über den Menüpunkt Zoom Bildschirmgröße können die Karten in das aktuelle Sichtfenster der Software gezoomt, d.h. die Karte wird komplett dargestellt. Der Menüpunkt Zoom Originalgröße zeigt die Karte wieder in Ihrer Originalgröße. Speziell beim Zoomen kann es zu Wartezeiten durch die Berechnung der neuen Abbildungen der Karten kommen. Der Menüpunkt *Zoom +* ist zusätzlich über die Tastenkombination STRG-(Tastenblock)+, der Menüpunkt *Zoom -* über die Tastenkombination STRG-(Tastenblock)- und der Menüpunkt *Zoom Originalgröße* über die Tastenkombination STRG-R erreichbar.

7.2.8. Menü Info

Im Menü Info sind allgemeine Informationen über das Projekt, das Programm, den Autor und der Version des Programms abrufbar.

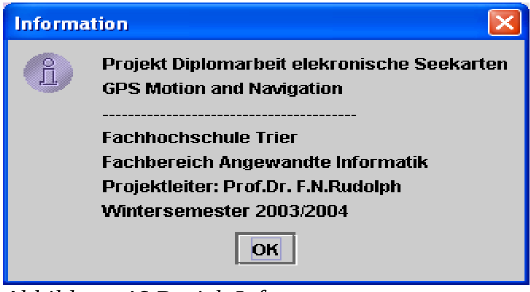


Abbildung 46 ProjektInfo

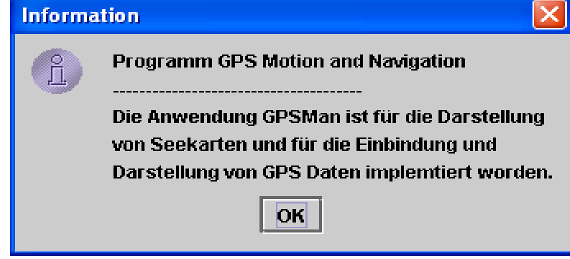


Abbildung 45 ProgrammInfo

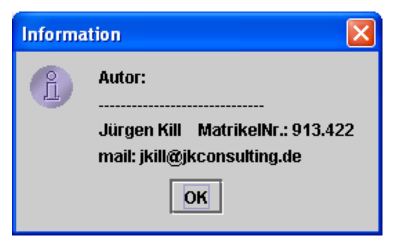


Abbildung 47 AutorenInfo



Abbildung 48 VersionInfo

7.3. Die Toolbar oder Buttonleiste

Die Toolbar ist als zusätzliche und schnellere Auswahlmöglichkeit für fast alle Menüpunkte vorgesehen worden. Hier eine Auflistung von links nach rechts der einzelnen Buttons in der Buttonleiste.

- Seekarten Dateien öffnen (*.KAP oder *.JPG)



Abbildung 49 Buttonleiste

- Zoom vergrößern
- Zoom verkleinern
- Originalgröße der Karte wiederherstellen
- Logbuch öffnen
- Neues Logbuch erstellen
- manuelles Kalibrieren von Seekarten im *.JPG Dateiformat
- Informationen über die Parameter einer *.KAP Datei
- Zeige die aktuelle Position des Schiffes
- Zeige den GPS Dialog
- Tracking Modus ein-/ausschalten
- Wegpunkte erstellen
- Wegpunkte verwalten
- Trackpunkte erzeugen (Simulation oder GPS)
- Trackpunkte verwalten
- Distanzmessung vornehmen
- Distanzmessungen auf der Seekarte löschen
- Kursberechnungen durchführen
- Kursberechnungen auf der Seekarte löschen
- Messergebnisfenster ein-/ausschalten bzw. zeigen
- Einstellungsfenster öffnen

Die Funktionalitäten bezüglich der Buttons sind schon in den einzelnen Kapiteln der Menüpunkte beschrieben worden. Nicht alle Menüpunkte sind als Buttons verfügbar, dies ist durch die beschränkte Größe der Buttonleiste bedingt. Hier wurden nur die wichtigsten Funktionen auch als Buttons in der Buttonleiste dargestellt.

7.4. Statuszeile

Die Statuszeile zeigt Informationen über die aktuelle Position des Mauszeiger auf der jeweiligen Seekarte.

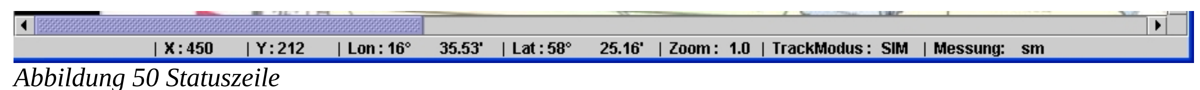


Abbildung 50 Statuszeile

Dargestelllt werden folgende Informationen:

- Die aktuellen Positionen der Bildpixel in der X- und Y-Koordinate
- Der Breiten-(Lat) und Längengrad(Lon)
- Der aktuell eingestellte Zoomfaktor
- Der derzeitige Simulationmodus (Simulation oder Echtdaten vom GPS)
- Die Maßeinheit die Distanzmessungen

8. Zusammenfassung und Ausblick

8.1. Zusammenfassung

In Anbetracht der Aufgabenstellung zu dieser Diplomarbeit, komme ich zu folgendem Fazit:

- Die Darstellungsgüte der Seekarten ist ausreichend für den praktischen Einsatz der Software

- Die Geschwindigkeit der Berechnungen (Simulation, Darstellung von Positionsdaten, Messungen, usw.) ist ebenfalls für den praktischen Einsatz der Software ausreichend.

- Die Plattformunabhängigkeit ist bereits durch die Programmiersprache Java vorgegeben und die Software wurde auch ausreichend unter den verschiedenen Betriebssystemen getestet.

- Die Problematik mit der nicht korrekt arbeitenden Kalibrierungsmethode bei großen Seekarten im *.KAP Format trat erst im Endstadium des Projektes auf, so dass keine Chance (im zeitlichen Rahmen gesehen) mehr bestand eine bessere und korrektere Kalibrierungsmethode zu entwickeln. Für die manuelle Kalibrierung von JPG Karten ist die implementierte Methode ausreichend, für die KAP Karten (die mehr als drei Referenzpunkte besitzen, z.B. bei großen Karten) ist ein anderer mathematischer Ansatz erforderlich.

- Die Geschwindigkeit zur Darstellung der Seekarten, im Besonderen beim Zoomen, ist **nicht** ausreichend. Die von der Progmmiersprache Java im Standard zur Verfügung gestellten Methoden sind für die Umrechung größerer Bilder scheinbar viel zu langsam. Es gibt jedoch weiterführende Klassen im Bereich der 2D Darstellung, die eventuell hier die entsprechende Geschwindigkeit zur Darstellung und Umrechnung bereitstellen. Eine Einarbeitung in diese Klassen liegt allerdings außerhalb des zeitlichen Rahmens dieser Diplomarbeit.

Somit lässt als abschliessender Satz sagen, das bis auf die Tatsache der sehr langsamen Umrechnung der Seekarten bei Zoomen und der Kalibrierung(wo das Problem nicht an dem grundsätzlichen System, sondern an einem anderen mathematischen Ansatz liegt), in allen anderen Bereichen ausreichende Ergebnisse für den praktischen Einsatz einer solchen Software erzielt wurden.

8.2. Anmerkungen

Bei genauer Betrachtung einiger eingebauten Mechanismen im Programm kann man feststellen, das hier einige Elemente aus früheren Projekten (Projekt SKKD) verwendet und auch wiederum verändert wurden. Dies hat sich aus der weiterführenden Einarbeitung in Java, sowie aus anderen zwecksmässigen Gründen so ergeben.

„**Softwareentwicklung ist ein iterativer Prozess**!“ Zitat aus dem Buch 'The UML User Guide'

Die Verweise auf Webseiten (siehe Literaturverzeichnis) wurden mit den Erstellern der jeweiligen Webseiten abgesprochen und eine entsprechende Erlaubnis zur Veröffentlichung einiger Bestandteile dieser Webseiten wurde vorher eingeholt.

9. Verzeichnisse

Literaturverzeichnis

IRL01: Thomas Irlbeck, Computerlexikon, 1998
LAB01: Otmar Labonde, Astronavigation, 2004, http://otmarlabonde.de
STU01: Rainer Stumpe, Informationen über nautische Navigation, 2004, http://www.rainerstumpe.de
BÖH01: Winfried Böhm, Handbuch der Navigation, 2003
CORE1: Cay S. Horstmann, Gary Cornell, core Java Grundlagen, 2001
CORE2: Cay S. Horstmann, Gary Cornell, core Java Expertenwissen, 2002
COREAWT: David M. Geary, core Java, die JFC behereschen (AWT), 1999
CORESWING: David M. Geary, core Java, die JFC beherrschen (Swing), 2000
Java-mitp: Aaron Walsh, John Fronckowiak, Java, Standardwerk, 2002
HÖH01: Rainer Höh, Orientierung mit Kompass und GPS, 2000
SCHW01: Wolfram Schwieder, Richtig Kartenlesen, 2001
BOR01: Malte Borges, Jörg Schumacher, StarOffice 6.0 (auch für OpenOffice), 2002
AST01: Frank Sauer, Besteckberechnung, 2003, http://www.astrosail.de
NAU01: Markus Eisenbart, Nautictools...einfach navigieren, Astronaut. Navigation, 2003, http://www.nautictools.de

Stichwortverzeichnis

Abbildungsverzeichnis

10. Anhang

Im Anhang zu der Diplomarbeit befinden sich folgende Bestandteile:

- Compact Disk (CD)
    - kompletter SourceCode der Diplomarbeit
    - Dokumentation der Diplomarbeit
        - im PDF Format
        - im OpenOffice Format
    - Java Dokumentation der Diplomarbeit
    - alle genutzten Dateien zur Diplomarbeit
    - alle genutzten Softwaretools zur Diplomarbeit